Application of PC-Based Project Management in an Integrated Planning Process

U. S. DEPARTMENT OF THE NAVY
CARDEROCK DIVISION OF
THE NAVAL SURFACE WARFARE CENTER

in cooperation with

Newport News Shipbuilding

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
MAY 1992	**N/A**	**-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Application of PC-Based Project Management in an Integrated Planning Process	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Surface Warfare Center CD Code 2230-Design Integration Tools Bldg 192, Room 128 9500 MacArthur Blvd, Bethesda, MD 20817-5700**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

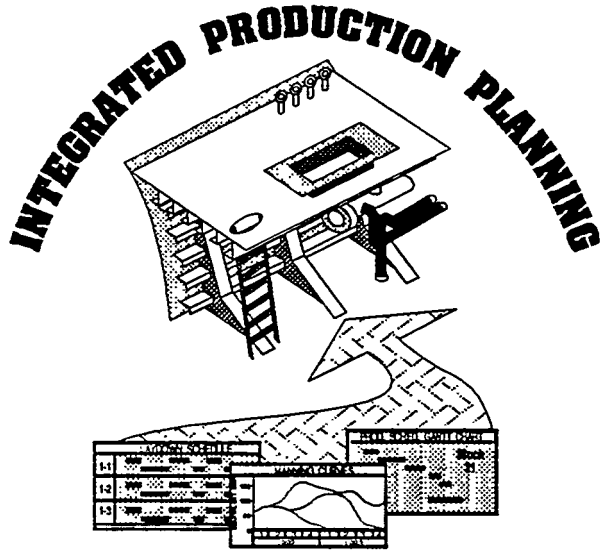
13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**SAR**	**199**	

APPLICATION OF PC-BASED PROJECT MANAGEMENT IN AN INTEGRATED PLANNING PROCESS



Prepared by
RICHARD J. NEUMANN
DAVID J. McQUAIDE

For
NEWPORT NEWS SHIPBUILDING AND DRYDOCK CO.

In behalf of
SNAME SHIP PRODUCTION COMMITTEE PANEL SP-8
on
INDUSTRIAL ENGINEERING

Under the
NATIONAL SHIPBUILDING RESEARCH PROGRAM

May 1992

ACKNOWLEDGEMENTS

This is the final report of a project developed and cost shared by National Steel and Shipbuilding Co. (NASSCO) under a David Taylor Research Center contract. The program is a cooperative effort of the U.S. Navy and the United States shipbuilding industry. Industry direction was provided by the Society of Naval Architects and Marine Engineers' (SNAME) Ship Production Committee on Industrial Engineering (SP-8). The program manager is W.G. Becker of Newport News Shipbuilding.

The authors would like to thank Jim Royle, Jon Gribskov, Pete Jaquith, Len Schneider and John Lyle of NASSCO for believing in our ideas and allowing us the time to develop the system. We would also like to thank Lyn Haumschilt, Bill Wilson, and many others at NASSCO for their assistance, insight, and advice without which this paper would not have been possible. Appendix VI of this submittal is based upon the insights gained by a NASSCO team headed by Steve Eckberg. Steve and his team deserve credit for the excellent analysis of the applicability of stochastic simulation vs. deterministic project management software in schedule development. We are indebted to Bill Caughlin and Welcom Software for their generous training, support, and recommendations. We would also like to thank the SP-8 panel for taking the time to review and comment on this submittal. Your inputs have made this a better paper.

PREFACE

The report and software included within this package were developed under the auspices of SNAME Ship Production Committee panel SP-8 Task N8-91-6, "Application of PC-Based Project Management in an Integrated Planning Process". The task was performed by Richard J. Neumann and David J. McQuaide of National Steel and Shipbuilding Co., San Diego. This submittal consists of four parts as shown below:

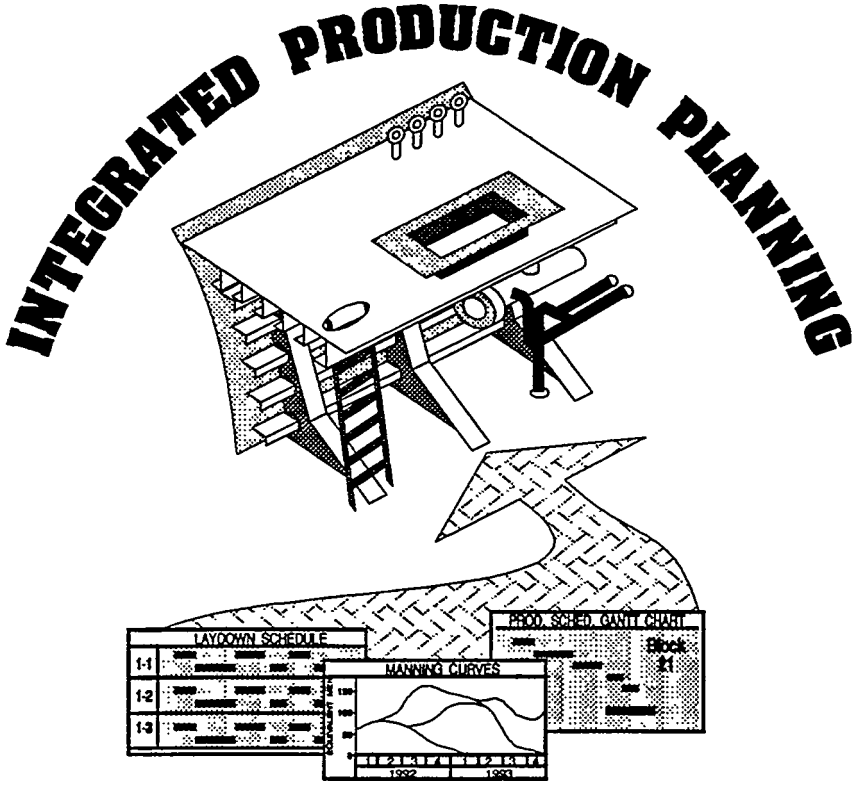
- The "Project Report" which describes the system's philosophy and overview.
- The system's "Users Manual" which explains the operation of the Integrated Production Planning System as developed in this package. The manual assumes the user has a moderate knowledge of dBase programming along with an understanding of project management techniques and ship production planning.
- An appendix containing program logic flow charts, program coding and documentation, and disk copies of all programs and data files developed by the project for use by the system.
- A "System Demonstration Disk" which is an on-screen slide show presentation where the user can step through the system and see the various screens of the Integrated Production Planning System along with on-screen explanations.

EXECUTIVE SUMMARY

This task has developed a PC-based system which serves as a tool to assist planning organizations in developing, updating, and revising ship production schedules. The system will also create and update manning, facility, and material utilization reports. The scope of the system developed is limited to the ground assembly, outfit, join, and erect operations. The "Project Report" describes the data required by the system to produce its outputs. The report explains the system development philosophy and gives an overview of the schedule generation system. To demonstrate the use of the system, data for a sample ship is given and a schedule developed based upon this data.

The User's Manual serves as a reference for shipyards wishing to develop a PC-based Integrated Production Planning System (IPPS). The software included in this package is not intended to be a turnkey system. For an IPPS to work for a particular shipyard, the shipyard must modify the coding so that the system will conform to the yard's facilities and methods of operation. The IPPS should not be viewed as a computer system; it is a production planning system that makes use of computer tools. Simply obtaining and installing the software will not give a shipyard an operable system. Developing an Integrated Production Planning System is a significant task. However, once developed, the IPPS is a valuable tool that will assist shipyard personnel in making effective production decisions.

PROJECT REPORT



SYSTEM

Project Report
Application of PC-Based Project Management
In an Integrated Planning Process

	Page
ABSTRACT	1
GLOSSARY	2
SCOPE OF PROJECT	3-4
SYSTEM DEVELOPMENT PHILOSOPHY	5-7
SCHEDULE GENERATION SYSTEM OVERVIEW	8-12
PROJECT MANAGEMENT SOFTWARE	13
STEPPING THROUGH THE SYSTEM	14-19
CONCLUSION	20
REFERENCES	20

ABSTRACT

When a schedule is developed for a project it dictates not only the dates when various activities should occur, but also the manning, facility, and material utilization required to meet the schedule. A change in schedule necessitates a change in manning, facility and material utilization plans. By the same token, changes in manning, facilities or material availability (e.g. late arrival of material) necessitates a change in schedule. Since the activities within a project are interrelated and various projects often use common manning and facilities, a change in a single activity often necessitates the modification of many related activities.

A shipyard working on several projects must schedule thousands of interrelated activities. To remain credible, the schedules must be updated as items are delayed and variations in production schedules occur. For a shipyard to remain competitive, it must have a production planning system that optimizes the yard's overall use of facilities and manning.

This paper discusses the development and implementation of a PC-based Integrated Production Planning System (IPPS) which serves as a tool to assist planning organizations in developing, updating, and revising schedules and associated manning, facility, and material utilization reports.

GLOSSARY

The following definitions are provided to clarify usage within this paper. They are not meant to imply any type of industry standard.

MASTER PRODUCTION SCHEDULE (MPS) - (as defined by the American Production and Inventory Control Society Dictionary) "... the anticipated build schedule for those selected items assigned to the master scheduler. . ." [1]

BLOCK- A structural assembly which will be outfitted and erected singly or as part of a grand block.

GRAND BLOCK- TWO or more blocks that have been joined into a single interim product prior to erection.

LONG RANGE FORECAST- Needs implied by schedules over a two to three year time span. This forecast will show manning and capacity needs of a project for its entire ground assembly, join, outfit, and erect period.

SHORT TERM REQUIREMENTS- Needs implied by schedules over a two to three month time span. Requirements are used for regularly updated, detailed manpower and facility utilization planning.

SCOPE OF PROJECT

An effective integrated production schedule will consider all activities that go on within a shipyard. However, this does not mean that a single production planning system must model all activities. If an individual system models a well-defined area of the shipyard, this information can be combined with information regarding other areas to develop an overall view of the shipyard system.

At the highest level, a shipyard may be described as consisting of four interrelated functions: production, materials, engineering, and personnel/ business administration/ business development. Production operations may be grouped in a variety of ways. The grouping used in this paper is as shown in figure 1.

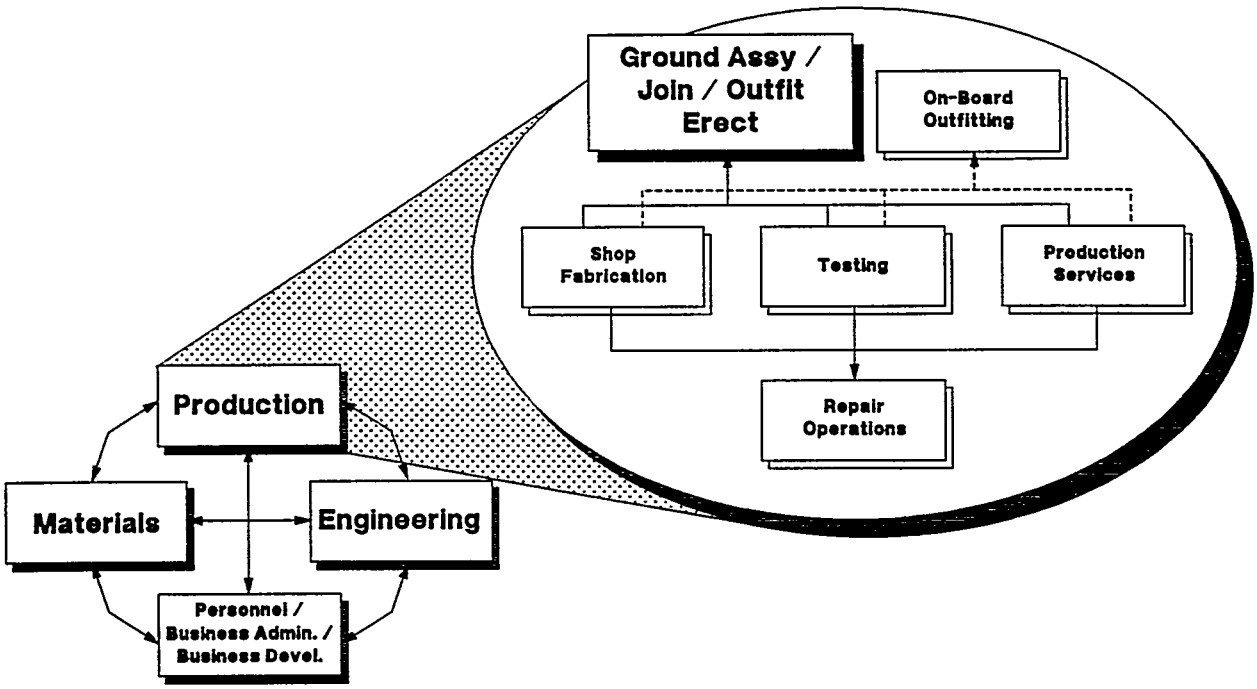


Figure 1: Interrelation of spyard activities.

The scope of activities to be modeled by the IPPS discussed in this paper are limited to ground assembly, joining, outfitting, and erection of blocks. The activities to be modeled are as listed.

- Fabrication of Steel Parts
- Block Sub-Assemblies (i.e., building of bulkheads, decks, etc., from fabricated parts)
- Block Assembly
- Pre-Blast Outfitting of Blocks/Grand Blocks
- Blast and Paint of Blocks/Grand Blocks
- Post-Blast Outfitting of Blocks/Grand Blocks
  Grand Blocking (joining of blocks before they erect to ship)
- Block or Grand Block Erection to Ship

The on-board outfitting, shop, production service, repair, and non-production activities are not modeled by this system. Schedules and information regarding these activities are developed in parallel with this system. The data is combined with data developed by the IPPS and is used to provide information regarding the entire shipyard.

The Master Production Schedule (MPS) must be coupled to bills of material structured to support the production process. They are not separate issues. A workable interface between the scheduling and material requirement system is vital. Development of this interface is dependent upon both the planning and material systems employed by the yard. This issue will not be addressed in this paper. However, when a production scheduling system is being developed, the scheduling/materials system interface must be considered.

The MPS also must be supported by engineering. Completion of engineering specifications and drawings must be scheduled to support the production and material ordering process. However, the scheduling of these items will not be considered within the scope of this project.

SYSTEM DEVELOPMENT PHILOSOPHY

The purpose of scheduling is to optimize the use of resources so that the overall production objectives are met. Scheduling involves the assignment of dates to specific tasks. Machine breakdowns, absenteeism, quality and performance problems, material shortages, and other factors complicate the ship building environment. Hence, the assignment of a date does not ensure that the work will be performed at that time. [2] A scheduling system should have the ability to adapt schedules to reflect changes in the ship building environment.

An effective model for use in production scheduling must reflect the strategy by which the ship will be built. These strategies establish the activity durations, resource utilization, and relationships to be used by the Integrated Production Planning System. Documents should be developed to describe the strategy by which the ground assembly, join, outfit, and erect process will occur. Table 1 shows five strategy sheets that, when taken together, will provide the information required to develop an effective Master Production Schedule for the process. (Note: the sheets are illustrated in the build strategy development section of the User's Manual.) All strategy sheets are reviewed, discussed, and approved prior to model development.

Even if a PC-based model of the production process is not developed, creation of the documents shown in table 1 is a valuable tool . By bringing together the various materials, engineering, production and support groups for the strategy review process, the build strategies and ship's design will often be substantially improved and subsequent changes will generally be reduced.

In addition to the strategy sheets, it is advantageous to develop a coding system for the Work Breakdown Structure (WBS) and the Organizational Breakdown Structure (OBS). Use of these codes enables the system to group its output in meaningful ways.

Schedule information is distributed to all required groups in a format meaningful to that group.

STRATEGY SHEET	DESCRIPTION
BLOCK BREAKDOWN DIAGRAM	IDENTIFIES BREAKDOWN OF SHIP INTO STRUCTURAL ASSEMBLIES AND SHOWS ASSEMBLIES THAT JOIN TOGETHER ('GRAND BLOCK') PRIOR TO ERECTION
INTEGRATED ASSEMBLY/OUTFIT STRATEGIES BY BLOCK TYPE	BLOCKS ARE GROUPED BY COMMON TYPE. ASSEMBLY/OUTFIT ACTIVITIES ARE DEFINED. DURATIONS, RESOURCE REQUIREMENTS AND GENERAL SCOPE OF WORK FOR EACH ACTIVITY ARE IDENTIFIED.
GRAND BLOCK STRATEGIES	A SPECIAL CASE OF THE INTEGRATED ASSEMBLY/OUTFIT STRATEGIES. IDENTIFIES WHICH BLOCKS ARE JOINED TOGETHER TO FORM GRAND BLOCKS AND THE METHOD BY WHICH THEY JOIN.
ERECTION "STAR" CHART	SHOWS THE DATE EACH ERECTABLE UNIT IS SCHEDULED TO BE JOINED TO THE SHIP.
PROCESS LANE STRATEGIES	SHOWS IN GANTT CHART FORMAT THE FLOW OF BLOCKS THROUGH EACH OF THE DEDICATED PROCESS LANES.

Table 1: Ground assembly, join, outfit, and erect strategy sheet descriptions.

A schedule dictates not only the dates on which various activities occur, but also a specific set of material, engineering , facility, and manning requirements. For a schedule to remain credible, it must account for actual material delivery, engineering drawing availability, facility availability, and manpower availability. This cyclic relationship implies that an achievable schedule can be derived only when these factors are considered together.

Data are facts concerning objects, events, relationships, and requirements. Information is data that has been organized in a form that is suitable for decision-making. The development of a

schedule in not an analytically complex task. Development is made complex due to the large volume of data which must be considered. The IPPS transforms the large volume of data which influences schedules into useful information. The clearest way to convey this information is through graphical displays of schedules, manning requirements, and facility utilization data. By showing relevant data in this graphical form, the system serves as a useful tool in generating and updating the MPS.

Based upon the above discussion, the required capabilities of an Integrated Production Planning System can be defined. The requirements for an effective system are shown in table 2.

REQUIREMENTS FOR AN INTEGRATED PRODUCTION PLANNING SYSTEM
Ž REFLECT A BUILD STRATEGY
Ž SHOW MATERIAL, ENGINEERING,FACIUIV AND MANNING REQUIREMENTS IMPLIED BY THE SCHEDULE
Ž DISPLAY ALL DATA IN A CLEAR MEANINGFUL WAY
Ž HIGH LEVEL OF FLEXIBILlTY AND ADAPTABILITY

Table 2: Production planning system requirements.

SCHEDULE GENERATION SYSTEM OVERVIEW

The flow chart in figure 2 shows the Integrated Production Planning Systems major inputs, outputs, and components. The system consists of four modules which interact to create both the baseline and regularly updated production schedules. The system also generates manning and facility long range and short term requirements. Each module is described in greater detail on the pages that follow.

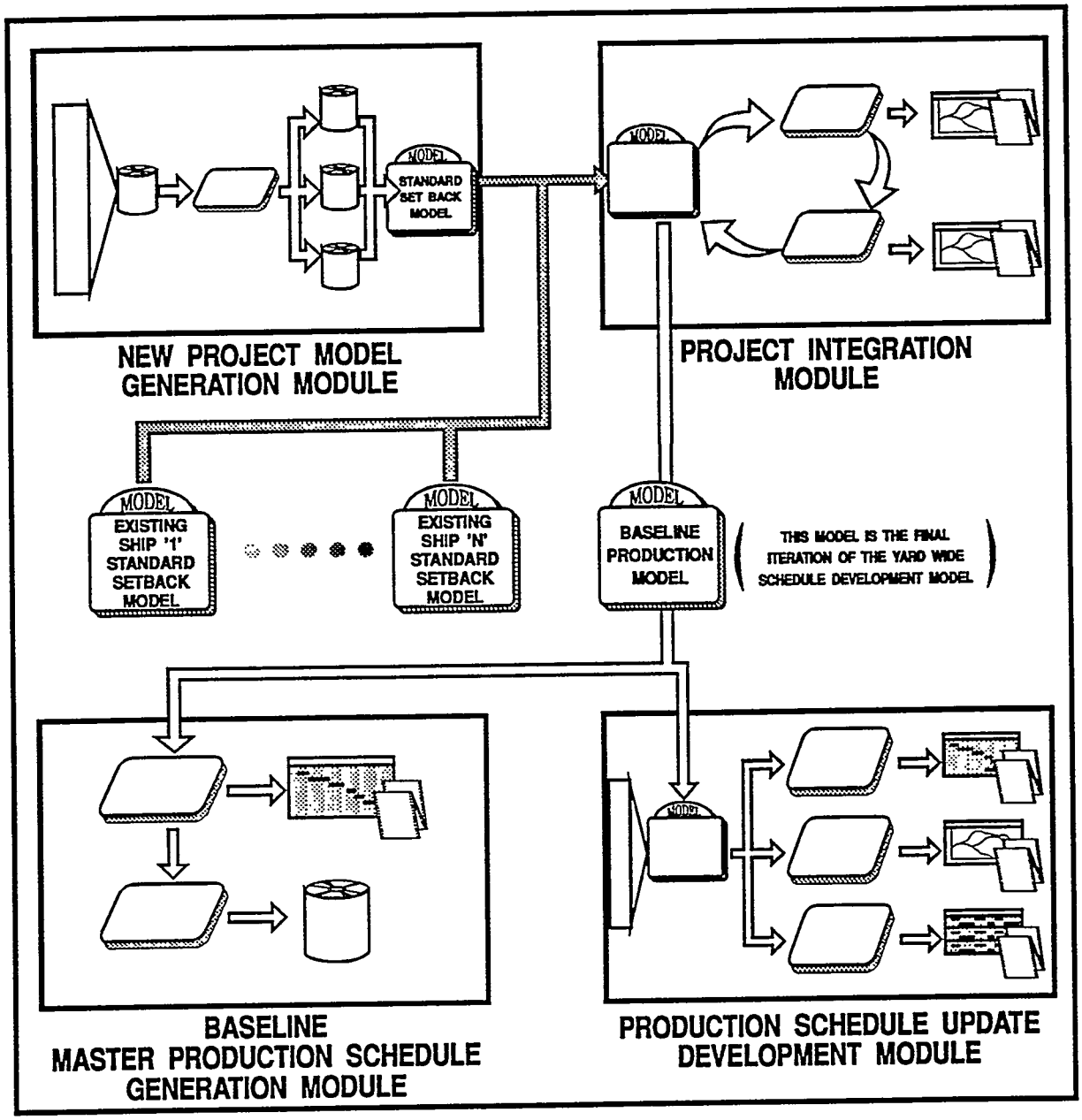


Figure 2: Integrated Production Pianning System flowchart

The New Project Model Generation Module is shown in figure 3. When a new project (i.e. a ship) is brought into the yard, data is gathered regarding build strategies, activity resource requirements, process lane considerations, and block erection data. All this data is collected into a Ship Build Master Data File. This data is passed through the Model Builder Program which creates a Standard Setback Model for the ship.

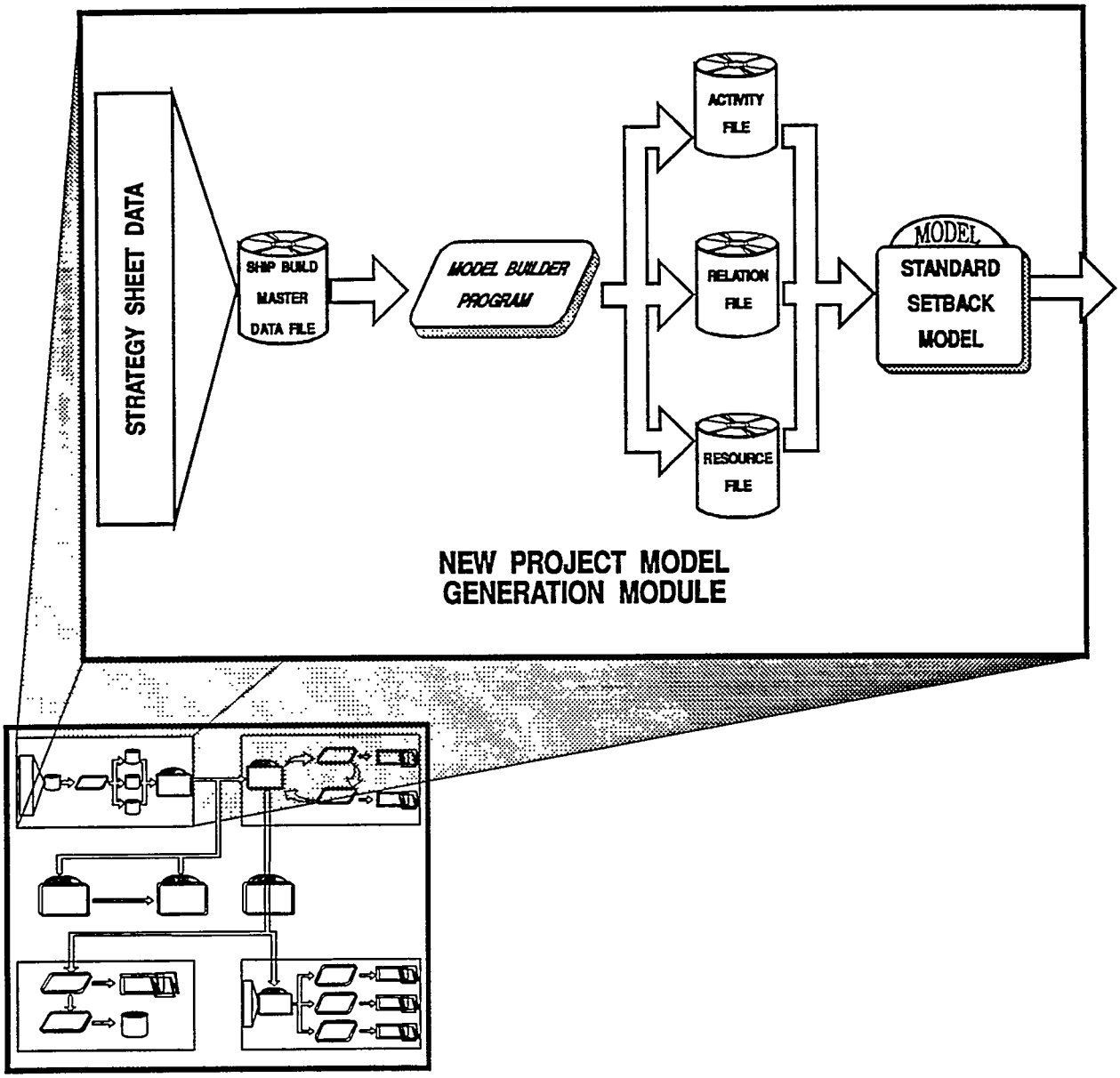


Figure 3: blew Project **Model** Generation Module.

The Project Integration Module is shown in figure 4. Standard Setback Models take into account build strategies for only the individual ship. Since all ships are built with common facilities and manpower, leveling the MPS is done with all the projects in the yard considered together. The Standard Setback Model is combined with the Standard Setback Models of previously scheduled work to create a Yardwide Schedule Development Model. This model is processed to show the capacity and manpower requirements implied by these schedules. Based upon this information, the model is refined through an iterative process of resource demand leveling. Capacities and manning implied by the schedule are investigated. Schedules are modified until acceptable capacity utilization and manning are achieved. The final iteration of the model is reviewed and approved by the various department heads. Upon approval, this model--becomes the Baseline Production Model.

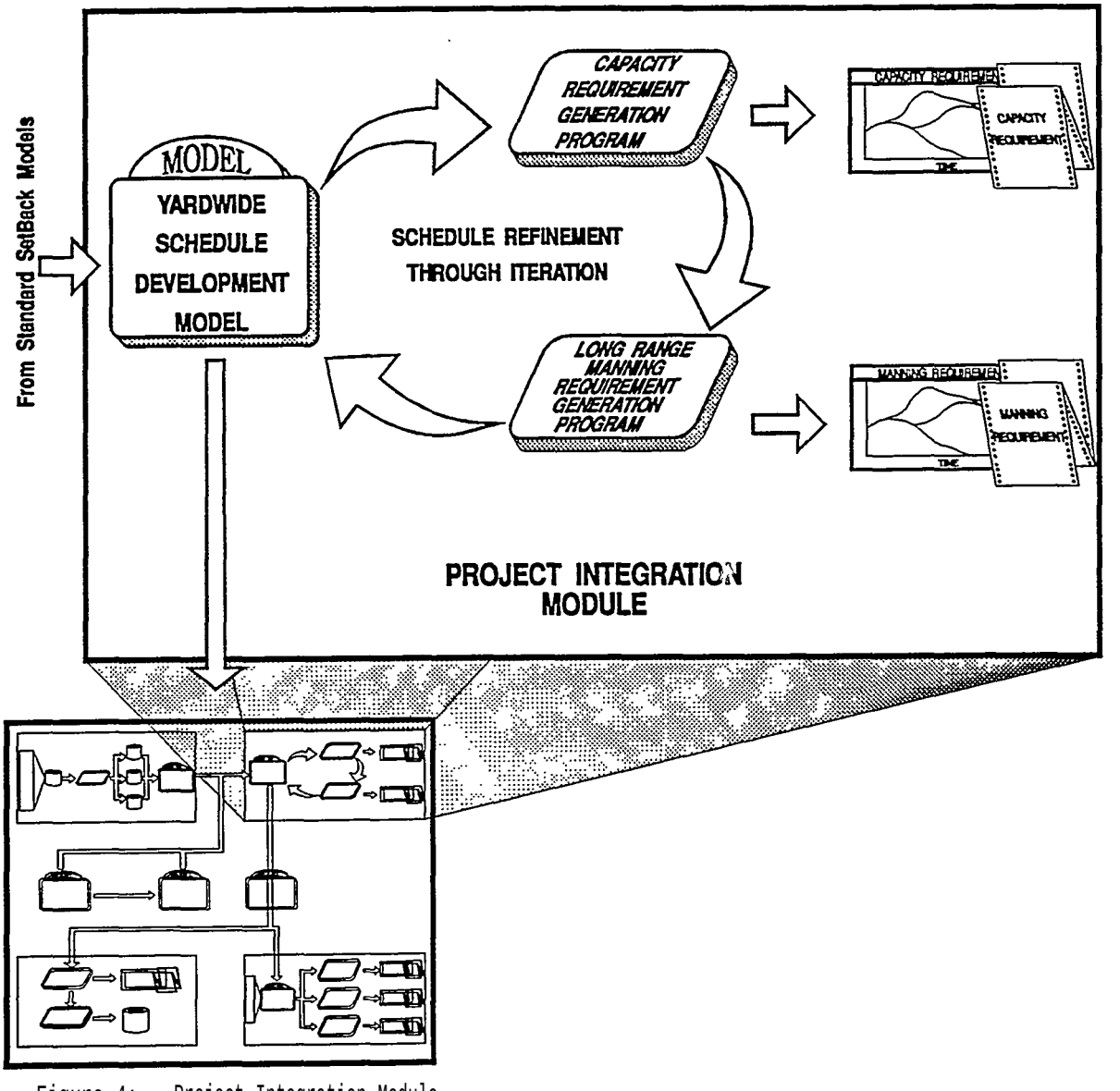


Figure 4: Project Integration Module.

The Baseline Master Production Schedule Generation Module is shown in figure 5. The Baseline Production Model is used to create the MPS for each project. The model is also used to create and update a database which serves as the baseline schedule for the schedule tracking system. The Baseline Production Model is altered only when a new revision of an existing projects schedule is issued.

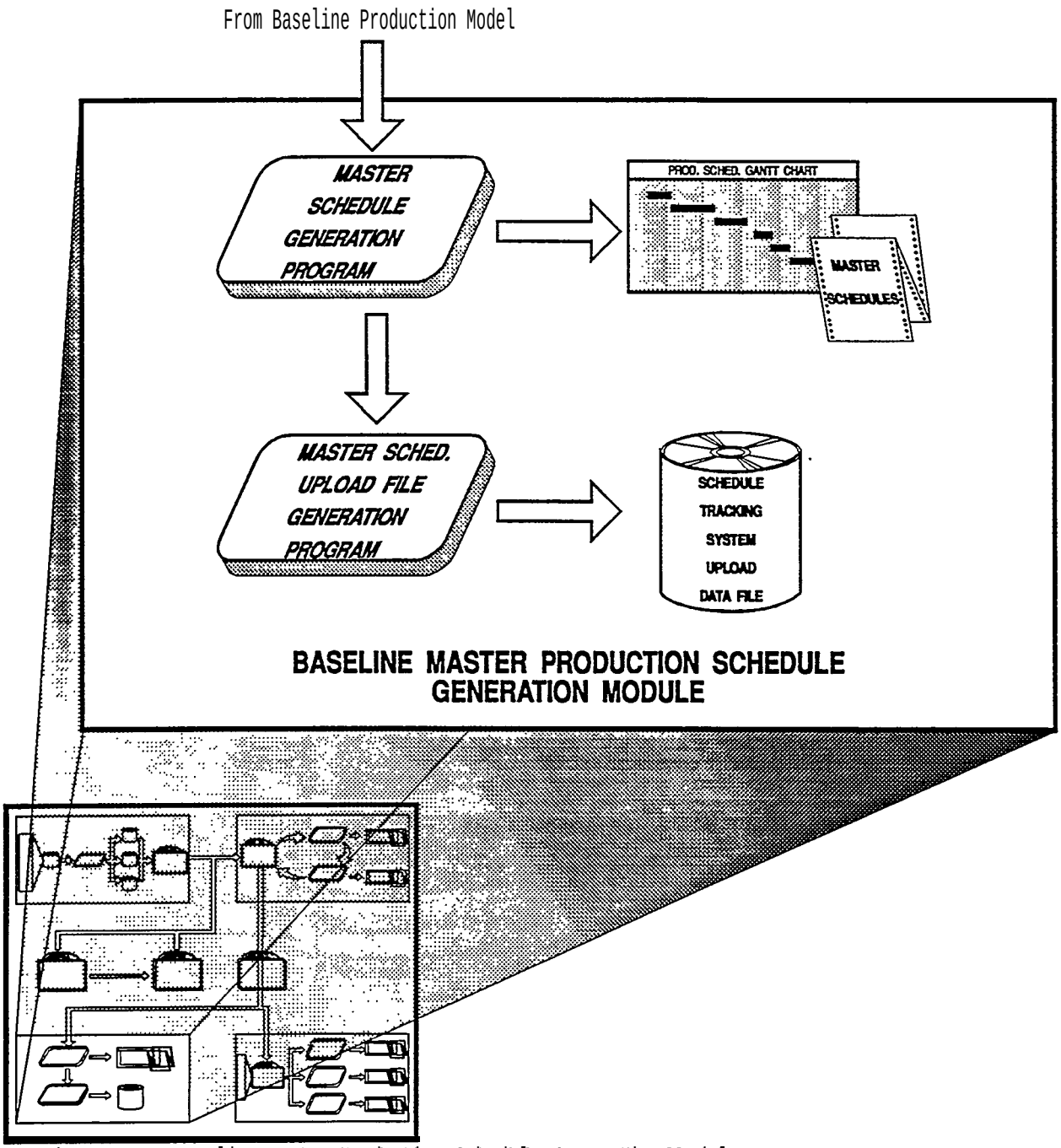


Figure 5: Baseline Master Production Schedule Generation Module

The Production Schedule Update Development Module is shown in figure 6. A copy of the Baseline Production Model is renamed the Production Update Model. This model is updated based on regular meetings and progress data. The updated model is processed and used to generate regularly issued production schedules, manning curves, and facility utilization reports (laydown schedules) . The manning curves and facility utilization reports reflect adjustments that have been made from the master schedule to the current production schedule.

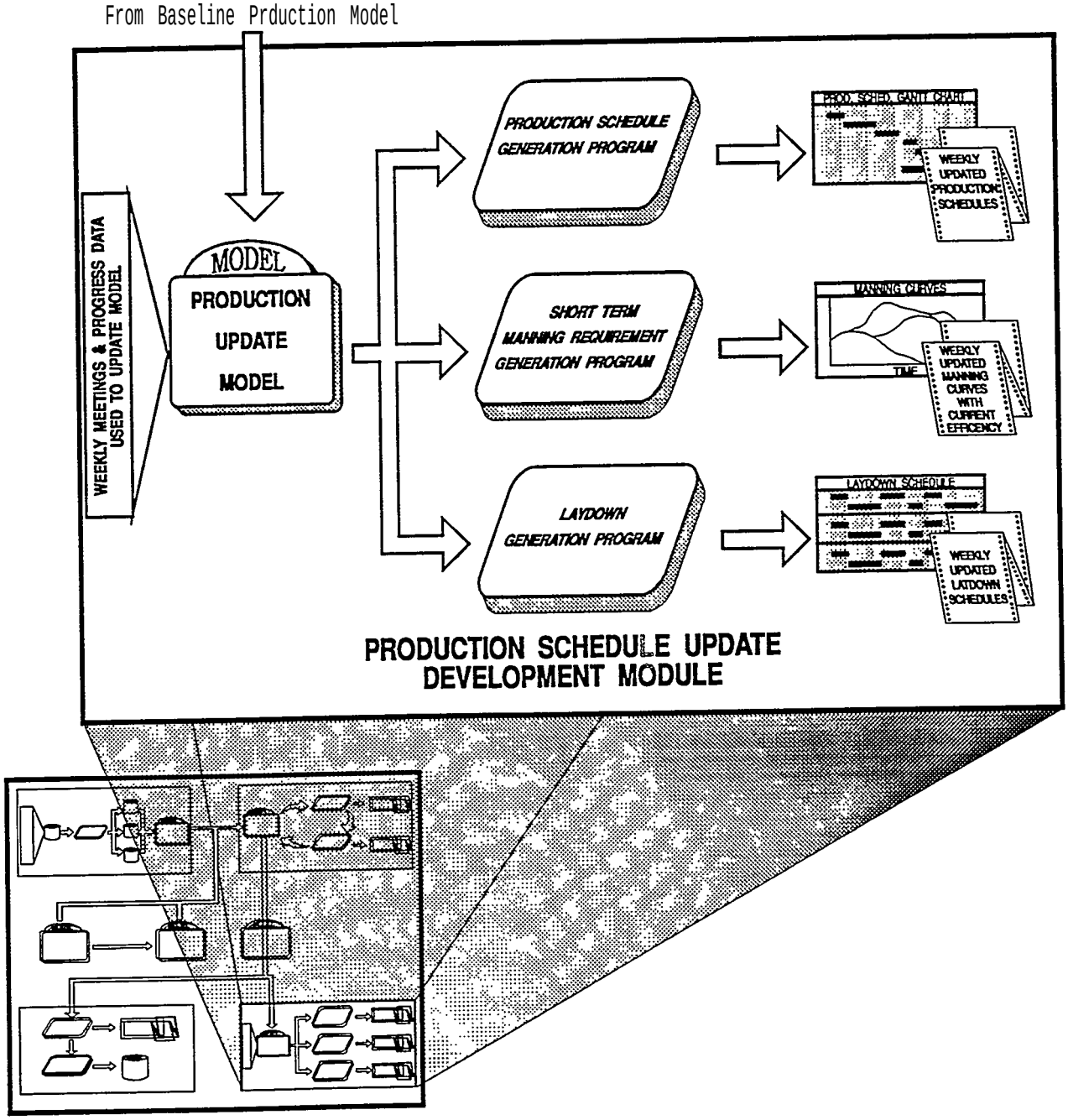


Figure 6: Production Schedule Update Development Module.

PROJECT MANAGEMENT SOFTWARE

The IPPS is built around Welcom Software Technology's Open Plan PC-based project management software . There are several PC-based project management packages on the market today. One of the advantages of Open Plan (herein referred to as the project management software) is that the software package operates within a dBase shell. All of the project management software input and output files are in standard dBase format. This allows all pre-processing and post-processing programs built around the project management software to be written in dBase.

All models shown in the Schedule Generation System Overview exist within the project management software framework. The software takes data regarding individual activities and creates schedules and resource utilization files. The data regarding individual activities is placed into three separate files. The activity file contains the duration of each activity. Before this file is processed, the only dates in the file are the start dates of key events (i.e. the date blocks or grand blocks erect to the ship) . The relationship file shows the required sequencing and interaction between various activities. The resource file shows the manning and facility requirements of each activity. The project management software processes these data files and generates all the dates that were not previously defined. These dates are stored to the processed activity file. The software also creates a resource aggregation file. This file shows the utilization of resources as a function of time. The major inputs and outputs of a project management software model are shown in figure 7.

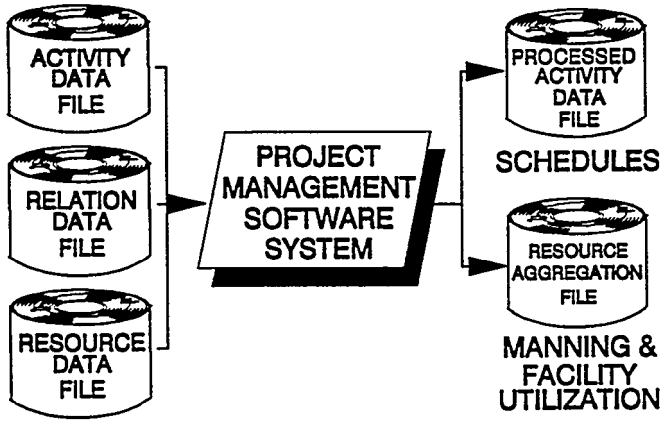


Figure 7: Project management software input and output files.

STEPPING THROUGH THE SYSTEM

To demonstrate how the IPPS is used, schedules were developed and updated for a test case. The test case is the construction of the MV Well Planned, a small, double-hulled product carrier. The first and most important task in scheduling is the development of a build strategy. The build strategy for the MV Well Planned is expressed in terms of the documents described in the Systems Development Philosophy section of this report. The strategy sheets for the MV Well Planned are shown in the Build Strategy Development section of the User's Manual(pages 6-10).

Information from the documents is used to create the MV Well Planned's Ship Build Master Data File. This file is processed by the Model Builder Program to create a Standard Setback Model for the ship. This model consists of the activities, resources, and relationships required to assemble, join, and outfit the blocks in preparation for erection. The erection activity for each block is fixed to a particular date as defined by the strategy sheets. Since the final event in each chain of activities is locked, the entire network of activities can be back-scheduled to show the late start and complete dates for each activity in the network.

The Standard Setback Model of the MV Well Planned shows the required start of construction date for the vessel to be 22 weeks before keel. This is not acceptable. To alleviate this early start of construction requirement, the build strategies must be altered. In the case of the MV Well Planned start of construction is driven by the wing tank block assembly process lane. To solve this problem the strategy was altered by using a second build position for this process lane. The revised process lane strategy sheet to reflect this change is shown in figure 8. The model is altered to reflect this new strategy by updating the relationships between the wing tank block assembly activities. The model is then reprocessed. The start of construction date with this new strategy becomes 10 weeks before keel. The build strategy is now

acceptable.

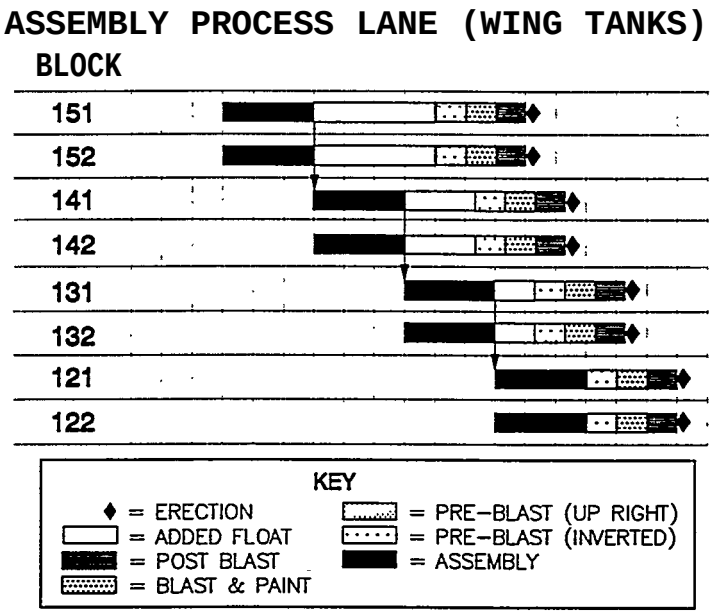


Figure 8: Revised process Iane strategy sheet.

The initial model is back-scheduled to late dates, therefore any leveling done is accomplished by moving activities earlier. Resource leveling strategies must reflect the constraints imposed by a particular yard's capabilities. If a yard has only a limited area to assemble the blocks, schedules may be leveled on space utilization within this area. If there is a required trade for which the yard has limited manning, schedules may be leveled based upon the trade's availability. Schedules may be leveled on any resource or combination of resources included within the model. Since the IPPS operates by back scheduling to late dates, the generalized resource leveling strategy is to first level resources in the area that immediately precedes the erection activity and then work back to earlier activities.

The strategy used for resource leveling of the MV Well Planned is to first level the outfit area manning. Since resources are interchangeable between the pre-blast inverted, pre-bast upright, and post blast outfitting activities, these activities are grouped by their common OBS code and leveled together. Once the outfitting

area is leveled, the assembly area is investigated. The assembly area in this example is leveled based upon the number of blocks with work in progress in both the flat and curved block build areas. By leveling first the outfitting area manning and then the assembly area work *in* process, a feasible MPS is created. This schedule reflects the build strategies for the vessel as well as taking into account the manning and facility availability.

Figure 9 shows the output of the Long Range Manning Requirement Generation Program for the outfit area. An analysis of the blocks outfitting during the peak months of April, May, and June show that the majority of the manning requirements during this period of time are driven by the outfitting of the house blocks and house grand blocks. In order to level manning in this area the outfitting of the lower house (grand block 531) will be scheduled prior to the outfitting of the upper house (grand block 533). To change the model to reflect this strategy a single relationship is added. The new relationship forces the outfitting of block 531 to complete before the outfitting of block 533 can start. Since the activities are linked, the system will reschedule the assembly, outfitting, and stacking activities of all blocks which comprise grand block 531. The results of this reschedule (iteration 2) are shown in figure 10.

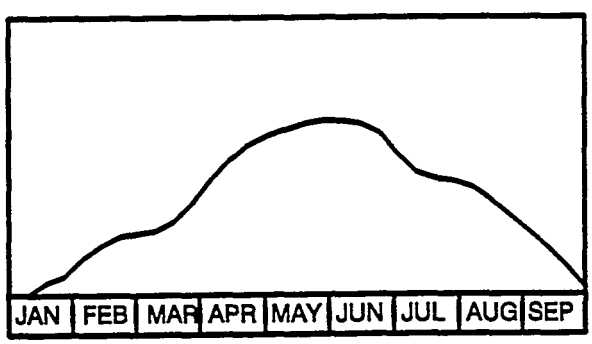


Figure 9: Outfit area manning requirements, iteration 1 - by standard setbacks.

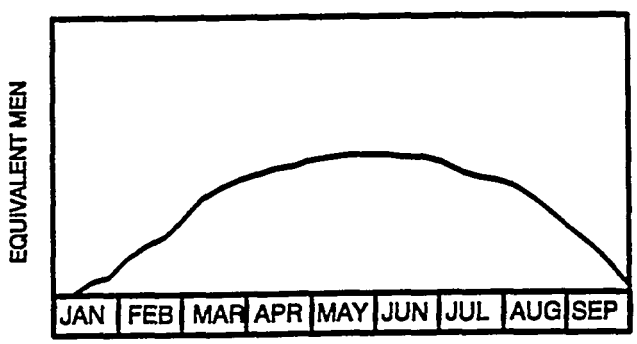


Figure 10: Outfit area manning requirements, iteration 2-grand block 531 forced early.

The manning curve for the outfitting area is now acceptable. Next, an analysis is made of the facility utilization within the assembly area. Two independent resources must be investigated within the assembly area. Both the flat block build platen and the shaped block build platen have limited space. The MPS must be adjusted so as to level both of these resources. Since the resources are independent, they may be leveled simultaneously.

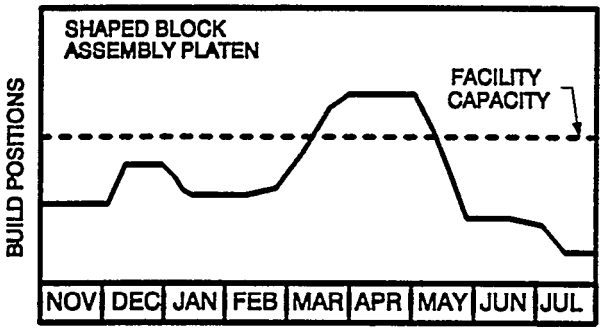


Figure 11: Shaped platen facility utilization based upon iteration 2 of the model.

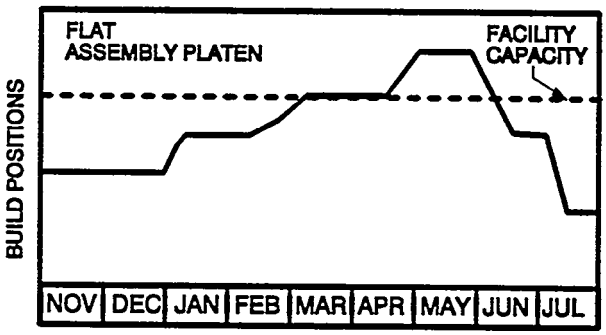


Figure 12: Flat platen faciliy utilization based upon iteration 2 of the model.

Figures 11 and 12 show the outputs of the Capacity Requirement Generation Program. Note that the system is back-scheduling to late dates and the assembly activity precedes the outfitting activities. Therefore, when assembly activities are forced earlier, float is introduced between the assembly and outfit operations. This has no impact on the outfitting area manning requirements. The resource leveling strategy for the MV Well Planned makes no attempt to level the assembly area manning. However, there is a high correlation between assembly build positions in use and assembly area manning requirements. If assembly build position usage is level, assembly area manning requirements are also fairly level. To level the shaped block assembly platen, some of the blocks scheduled to assemble in April are rescheduled to assemble earlier to fill in the valley in the February-March time period. When leveling the flat block assembly platen it is not desirable to take the excessive work in May and reschedule it for February. This would break the logical build sequence for the ship. Instead, the schedule should be modified to push earlier the building of a few blocks in March, April, and May.

This will eliminate the excessive capacity requirements while maintaining a proper build sequence. The results of this rescheduling (iteration 3) are shown in figure 13 and 14.

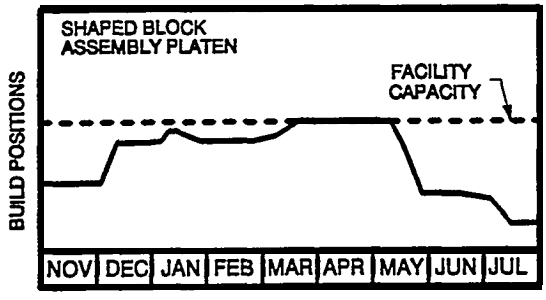


Figure 13: Shaped platen facility utilization based upon iteration 3 of the model.

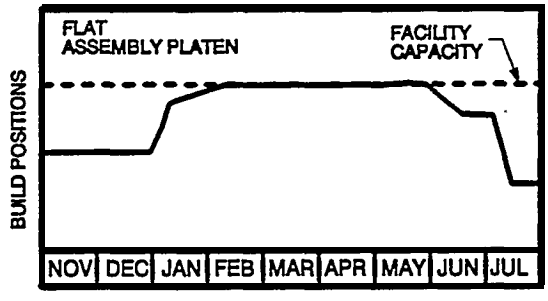


Figure 14: Flat platen facility utilization based upon iteration 3 of the model.

The facility utilization within the assembly area is now acceptable. This model is named the Baseline Production Model and processed by the Master Schedule Generation Program to create a Master Production Schedule. The MPS is approved by production, engineering, materials and support groups and the schedule is issued. An upload file is created to support the schedule tracking system.

A copy of the Baseline Production Schedule is renamed the Production Update Model. This model is updated based upon regular meeting and progress data. These weekly meetings are attended by members of the assembly, outfitting, erection, and support groups. These meetings serve to update the short term schedule documents based upon actual and projected progress.

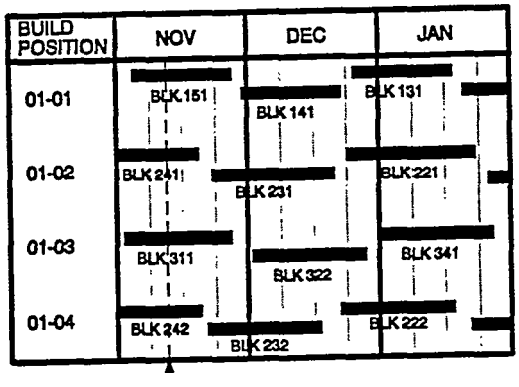


Figure 15: Flat platen assembly laydown chart

Figure 15 shows the current production schedule for the flat platen assembly area. At the production update meeting, the assembly area representative will report on actual and projected progress. The assembly area laydown chart is marked up by the assembly area representative as shown in figure 16. Based on the assembly area inputs, changes will be made to the production schedule. These changes are shown in table 3. The Production Update Model is modified to reflect the actions taken in the meeting. The model is then reprocessed and updated production schedules are issued to appropriate groups.

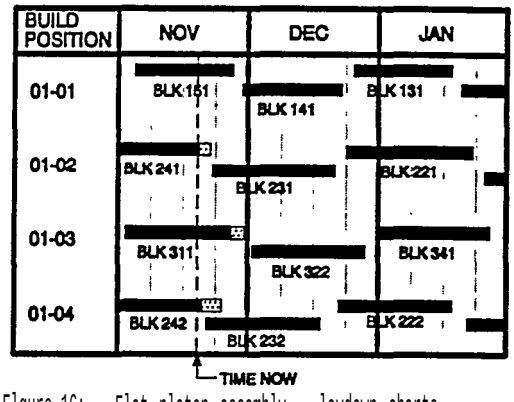


Figure 16: Flat platen assembly laydown charts modified by the assembly area rep. at the production update meeting.

CONFLICT	ADJUSTMENT
BLOCK 2421 IN BUILD POSITION 1-4 WILL OVERLAP WITH LAYDOWN OF BLOCK 232.	ASSEMBLY AGREES TO CRASH THE DURATION OF BLOCK 232 IN ORDER TO RECOVER TO THE SCHEDULE.
THE EXTENDED DURATION OF BLOCK 311 IN BUILD POSITION 1-3 DOES NO CAUSE A CONFLICT IN THE ASSEMBLY AREA. HOWEVER, THIS 3 DAY DELAY WILL CUT INTO THE SCHEDULED OUT FlTTING DURATION,	OUTFITTING,MADE AWARE OF THE DELAY AND ITS IMPACT AHEAD OF TIME, AGREES TO WORK THIS BLOCK MORE AGGRESSIVELY TO MAKE UP FOR THE DECREASED DURATION,
THE EXTENDED DURATION OF BLOCK 241 IN BUILD POSITION 1-2 HAS NO IMPACT ON OTHER BLOCKS IN THE ASSEMBLY AREA. THERE IS FLOAT BETWEEN THIS ACTIVITY AND Outfitting ACTIVITIES OF THIS BLOCK	LATER ACTIVITIES ARE NOT AFFECTED, THEREFORE, THIS DELAY HAS NO IMPACT.

Table 3: Adjustments made to the production schedule.

CONCLUSION

An effective MPS must reflect not only a build strategy, but also material, engineering, facility, and manpower availability. A PC-based system can be established to assist in creating and updating the MPS.

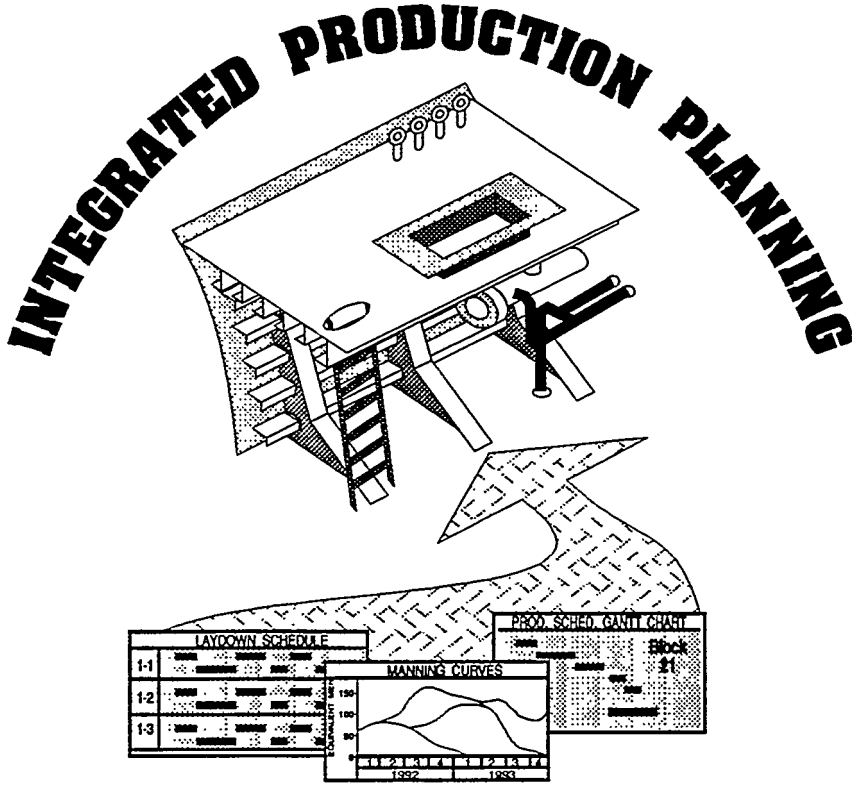
The system described in this reference will allow shipyard personnel to see the production schedule in a clear, meaningful way. This system can serve as a tool to help a shipyard make better production decisions and operate with improved effectiveness.

REFERENCES

1. APICS Dictionary, 5th ed., T.F. Wallace, Ed., American Production and Inventory Control Society, Falls Church, Va. 1984.

2. Production Planning and Inventory Control, D. McLeavey and S. Narasimhan, Allyn and Bacon, Inc. Newton, Mass, 1985.

USER'S MANUAL



SYSTEM

USER'S MANUAL
INTEGRATED PRODUCTION PLANNING SYSTEM

Page

PREFACE 1

SYSTEM SET UP

SOFTWARE/HARDWARE REQUIREMENTS. 2

SOFTWARE INSTALLATION 3

SYSTEM USE

SYSTEM INFORMATION REQUIREMENTS. 4-13

PROJECT CREATION 13-23

SYSTEM'S FILE REQUIREMENTS AND STRUCTURE

MODEL BUILDER PROGRAM

MODEL BUILDER DATA FILE

MODEL CREATION

SCHEDULE DEVELOPMENT. 24-30

PROCESSING THE MODEL

DEVELOPING THE LAYDOWN CHARTS

DEVELOPING RESOURCE UTILIZATION CURVES

DEVELOPING GANTT CHARTS

PROGRESSING AND UPDATING THE MODEL 31

CUSTOMIZING THE SYSTEM. 31

HELP 32

PREFACE

This User's Manual is provided as a reference for shipyards wishing to develop a PC-based Integrated Production Planning System (IPPS) . The software included in this package is not intended to be a turnkey system. To effectively implement a system, a shipyard should establish an IPPS development team. The team will define what that particular yard's needs are in an IPPS. They will also compile the ship's build strategy in proper format for use by the system.

This manual assumes that the development team has a moderate knowledge of dBase programming and an understanding of project management techniques and ship production planning. The user's group must also become familiar with the Open Plan software system.

The coding included with this package will operate for the sample organization and ship described herein. This package is intended as a skeletal outline showing how such a system can be established. For an IPPS to work at a particular shipyard, the yard must modify the coding so that the system will conform to the yard's facilities and method of operation. The coding and documentation provided is a base from which customized programs can be written for a yard's specific application. In developing and using the IPPS, the development team needs to have access to the following:

. This User's Manual
. An MS-DOS System
. Open Plan User's Manuals
. The Software and Hardware Described in the System Set-Up Section Which Follows

SYSTEM SET-UP

software/Hardware Requirements

The IPPS is developed to run on IBM compatible MS-DOS systems. The data files created when using this system will contain thousands to tens of thousands of records. For ease of use, a high performance 386 or 486 machine is recommended so that the programs will run in a reasonable time frame. The system as laid out in this report requires three pieces of commercially available software. The project management software selected is Open Plan. Open Plan operates within a dBase or FoxPro environment, so one of these pieces of software is also required. All coding included within this package is written in dBase. Manning curves generated by the system can be created by any graphics program that can chart an imported dBase file. The system described in this report makes use of Lotus Freelance. Points of contact for required software are shown in table 1.

PRODUCT	DEVELOPER	POINT OF CONTACT
Open Plan	Welcom Software	WST corporation 15995 North Barkers Landing, Suite 275 Huston, Tx77079 Tel: (713)558-0514
dBaee III	Borland International	Local softvvare dealer
Fox Pro	Fox Sofhvare	Local software dealer
Freelance	Lotus Development	Local software dealer

Tablel: Software requirements for the IPPS,

Software Installation Procedure

To install the IPPS programs and files onto your hard drive:

1) Open Plan must be installed and running on your computer.
2) Insert "Program Disk" Disk 1 (included in this project) into Drive "B"
3) Use the DOS COPY Command to copy all files from the "Program Diskl" Disk 1 into the Open Plan Directory. This is done by typing the statement shown below after prompt.

   C:> COPY B:*.* C:\OPLAN
4) Repeat the copy process for the "Sample Projects" Disk 2 (included in this package). Open Plan usually has a separate sub-directory for each of its projects, so make sure you enter the proper path. This is done by typing the statement shown below after the prompt.

   C:> COPY B:*.* C:\OPLAN\PROJ
5) When in Open Plan you must add the two sample projects, "MODEL" and "MODE", to Open Pian's Project Directory. Then use the Re-Build Index Utility to re-index the project directory and both projects.

Figure 1 lists all of the files copied from both the Program Disk and Sample Projects Disk.

Note: MODEL is the empty project with all necessary changes to the file structures. MODELA is a completely built project with table assignments for area laydown requirements. This model was created by processing the Build.dbf data file through the Model Builder Program. The standard setback model is then level loaded by both area and manning utilization.

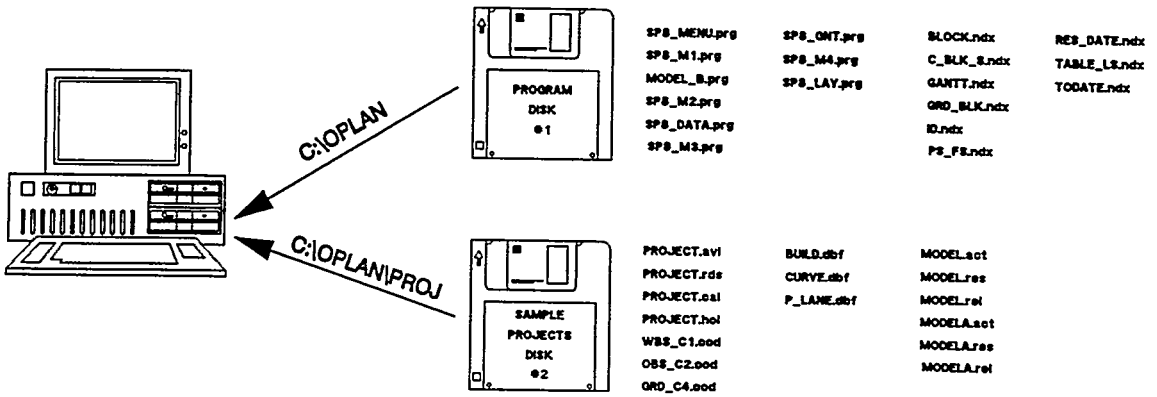


Figure 1: Installing the Integrated ProductionPlanning System.

SYSTEM USE

System Information Requirements

An effective model for use in production scheduling must reflect the strategy by which the ship will be built. This section will discuss what information should be developed and illustrates sample formats for gathering this information.

Documents may be developed to describe the strategy by which the ground assembly, join, outfit, and erect process will occur. Table 1 of the Written Report shows five strategy sheets that, when taken together, will provide the information required to develop an effective MPS for the process. Figure 2 below shows the strategy development logic flowchart for the creation of the Model Builder Data Base.

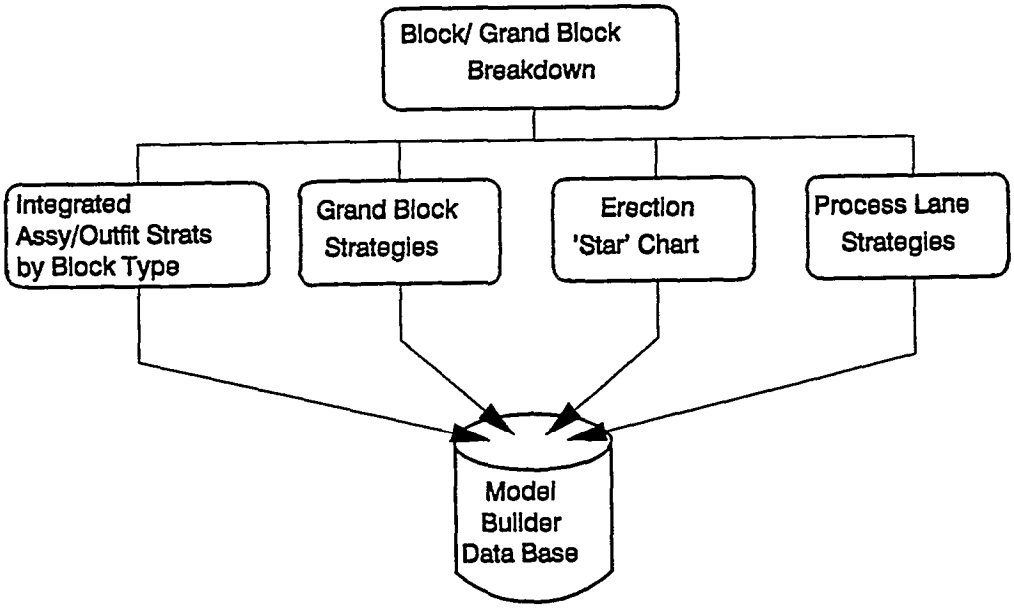


Figure 2: Strategy development logic flow.

The first step in developing this information is to create the block/grand block breakdown for the ship. This breakdown (shown in figure 3) drives the pre-outfitting strategy for the vessel. Once the block/grand block breakdown has been established, the other build strategy sheets may be developed. The ground assembly, join, outfit, and erect strategy sheets are shown in figures 4 through 7. An effective build strategy is a function of a yards facilities, capacities, and ship design. These documents should be developed through the joint effort of production, planning, materials, and engineering. This development should be done concurrently with the development of structural drawings and composites so that these documents can both influence and be influenced by the development of the block breakdown and build strategy.

BLOCK/GRAND BLOCK BREAKDOWN

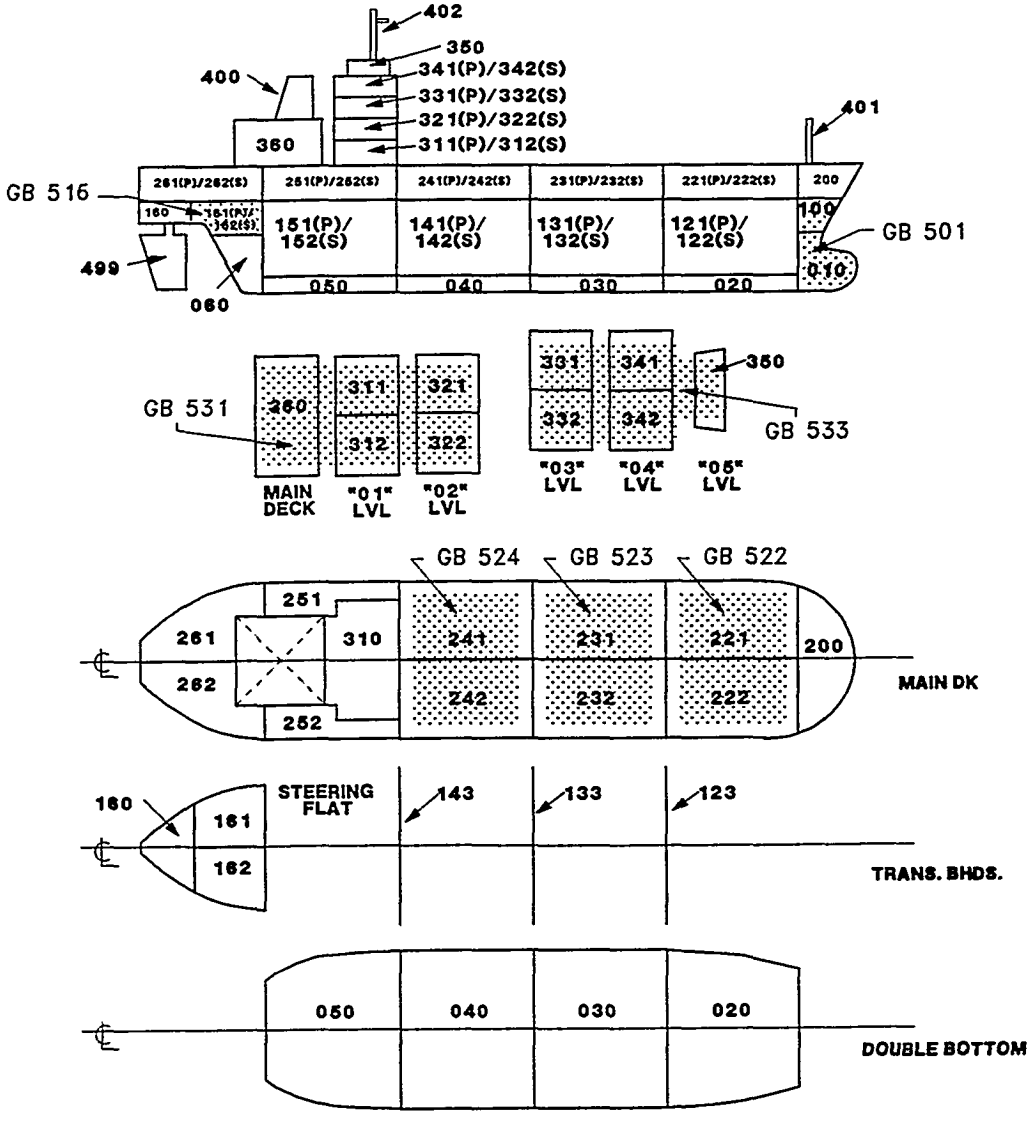


Figure 3: Block/Grand Block Breakdown for the MV WELL PLANNED. This figure shows the manner in which the vessel will be broken down into modules for construction. The figure also shows which of these modular blocks are joined together or "Grand Blocked" prior to their erection onto the ship. Each Block and Grand Block is assigned *a* unique name (a 3 digit number) so that the blocks may be identified and scheduled by the Integrated Production Planning System.

ERECTION "STAR" CHART

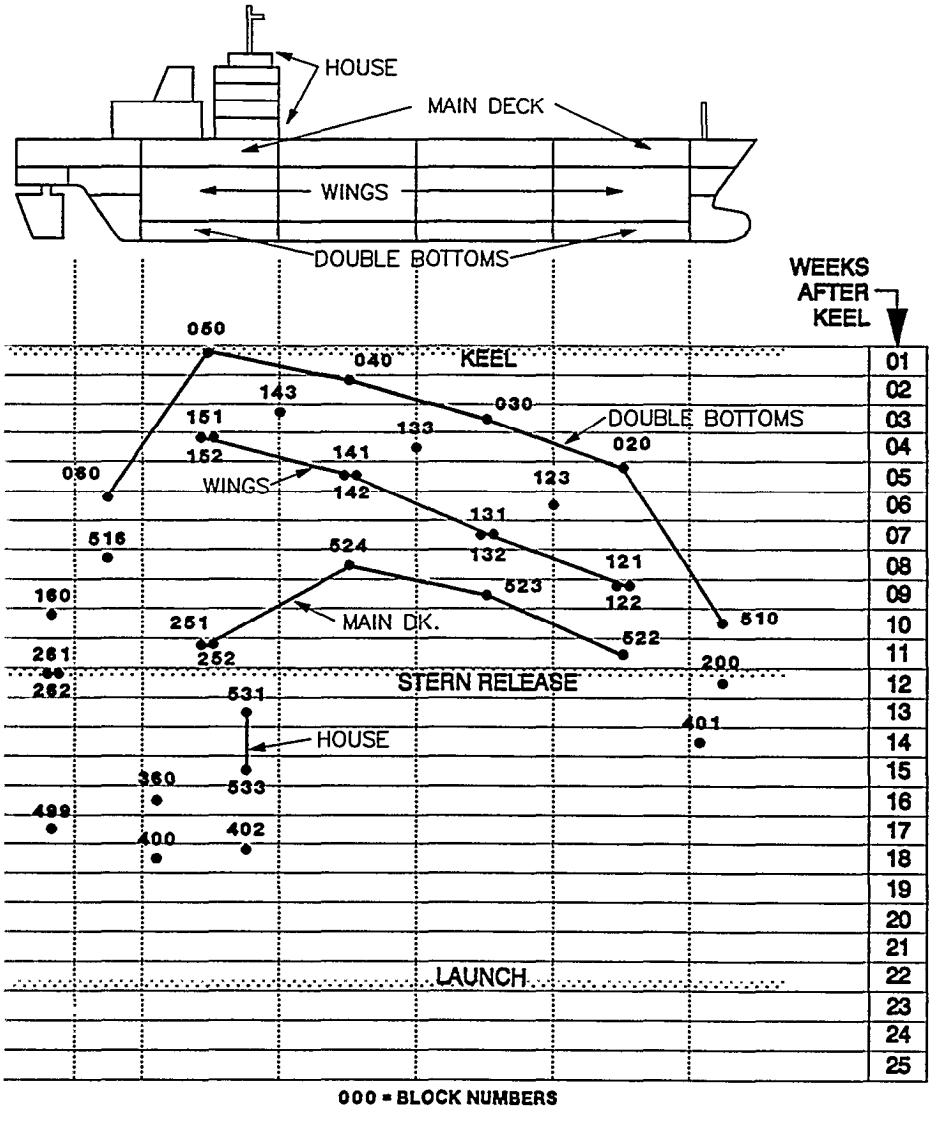
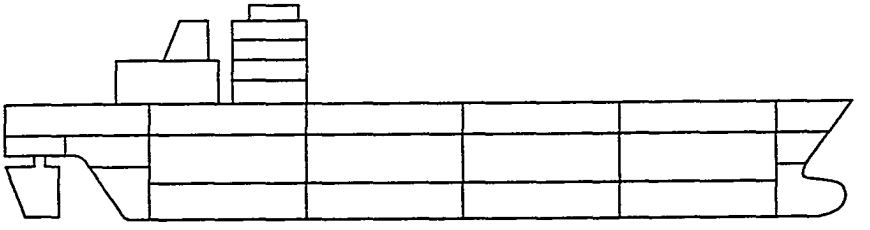


Figure 4 : Erection "star" chart for the MV WELL PLANNED.

This figure shows the date that each erectable block and grand block will be brought to the ship. These dates are the only ones that are initially fixed within the Integrated Production Planning System. All other dates (including the stack dates of blocks that go into grand blocks) are developed through the IPPS.

INTEGRATED ASSEMBLY/OUTFIT STRATEGIES BY BLOCK TYPE



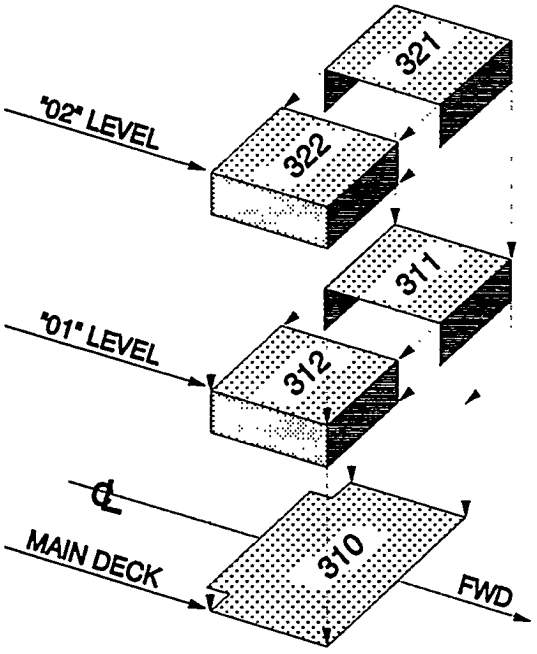
BLK TYPE	BLOCK NUMBERS	BUILD STRAT.	ASSY	PRE-BLAST INVERTED	PRE-BLAST UP RIGHT	BLAST & PAINT	POST BLAST	ERECT/ STACK
DOUBLE BOTTOM	020,030 040,050	STANDARD	20 1500	-	5 250	5 100	5 250	20 1000
TRANS. BHD.	123,133 143	STANDARD	15 500	5 150	-	5 100	5 150	20 400
STERN	160,261 262	STANDARD	15 2000	10 300	5 150	5 100	5 150	20 1500
STERN	161,162	GRAND2	15 1000	-	-	-	-	20 200
MAIN DECK	251,252	STANDARD	15 800	5 200	-	5 100	5 200	20 500
MAIN DECK	221→242	GRAND4	15 1000	-	-	-	-	20 150
WING	121→152	STANDARD	15 800	5 150	-	5 100	5 150	20 600
BOW	200	STANDARD	20 1500	5 125	5 125	5 100	10 125	20 1000
BOW	010,100	GRAND3	20 2000	-	-	-	-	20 350
HOUSE	310→350	GRAND1	15 800	20 800	5 200	5 100	-	20 300
MAST	401,402	SHOP	-	10 300	-	5 100	20 700	20 350
RUDDER	499	STANDARD	30 1000	-	-	5 100	-	20 650
CASEING	360	STANDARD	15 900	10 500	10 500	5 100	10 500	20 650
STACK	400	STANDARD	20 1000	-	10 500	5 100	10 500	20 600

ACTIVITY DURATION

ACTIVITY MANPOWER REQUIREMENT

FIGURE 5: Integrated Assembly/Outfit Strategies for the MV WELL PLANNED. This figure shows the build strategy activity durations in days, and manning requirements in hours for each of the block making up the ship. This table is developed jointly between steel and outfit planners and production personnel. Durations and budgets shown are sufficient to accomplish both the structural and the outfitting work planned in each step. Work should be scheduled so that the optimum amount of outfitting is installed on the block prior to erection. Strategies should also consider the sequencing of structural and outfitting work so that they proceed in an effective manner.

GRAND BLOCK STRATEGY
LOWER HOUSE - GB 531



BLOCK	GRAND	FIRST STACKING BLOCK	STACKING LAG FROM FIRST BLOCK
310	531	310	0
311	531	310	10
312	531	310	10
321	531	310	20
322	531	310	20

BLK TYPE	BLOCK NUMBER	GRAND	BUILD STRAT.	ASSY	PRE-BLAST INVERTED	PRE-BLAST UP RIGHT	BLAST & PAINT	POST BLAST	ERECT/ STACK
HOUSE GB	531	531	GRAND2	—	—	40 2500	5 700	15 1500	20 1200

Figure 6: Grand Block Strategy for the lower house of the MV WELL PLANNED. This figure shows the build strategy, activity durations, and manning requirements for the lower house (GB 531). The figure shows which blocks comprise this grand block and the time phasing by which these blocks stack into the grand block. A sheet such as this is prepared for each of the vessel's grand blocks.

PROCESS LANE STRATEGIES

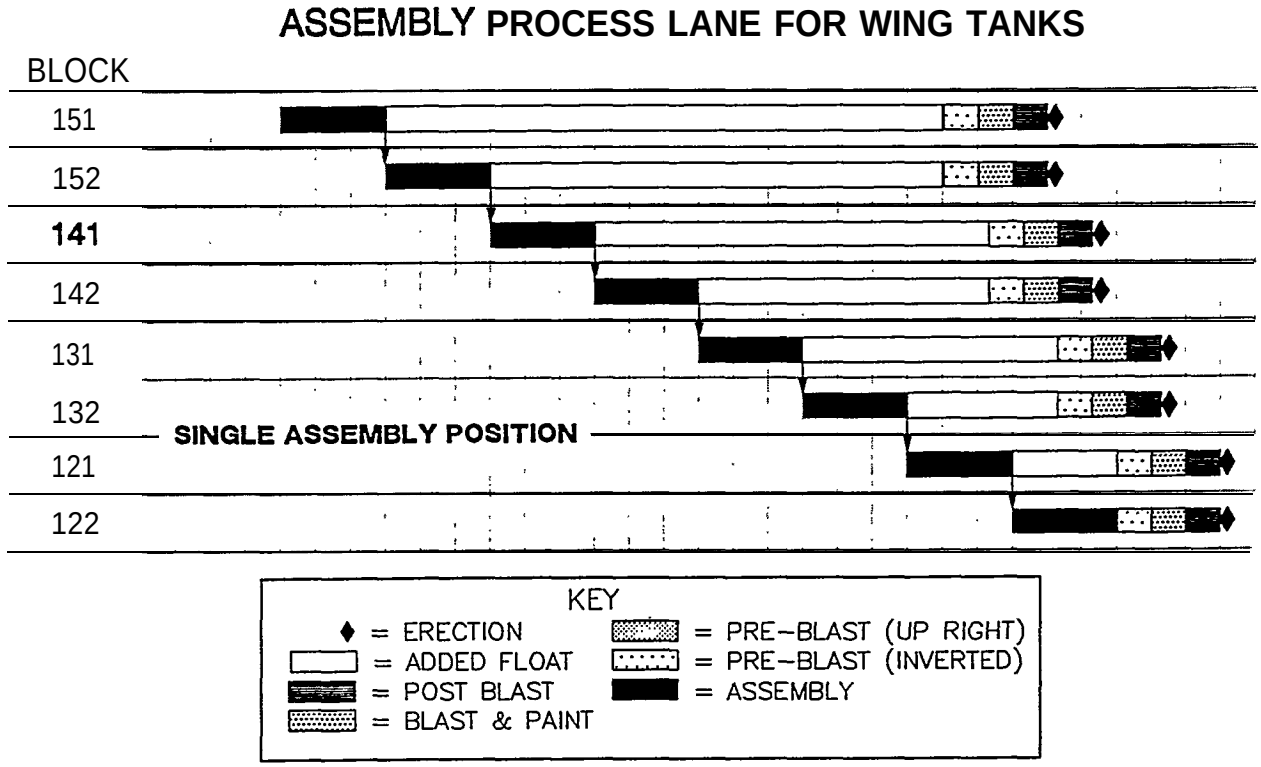


Figure 7: Process Lane Strategy for the MV WELL PLANNED. The figure shows a Gantt chart for production of wing tanks. The assembly process for each of these blocks will occur in a single dedicated work position. In order to accommodate this, float is introduced into the schedule of the early blocks through the process lane.

As noted in the System Development Philosophy section of the Project Report, a Work Breakdown Structure (WBS) and Organizational Breakdown Structure (OBS) should be created so that all data developed by the system may be viewed and aggregated in meaningful ways. These structures are shown in figures 8 and 9.

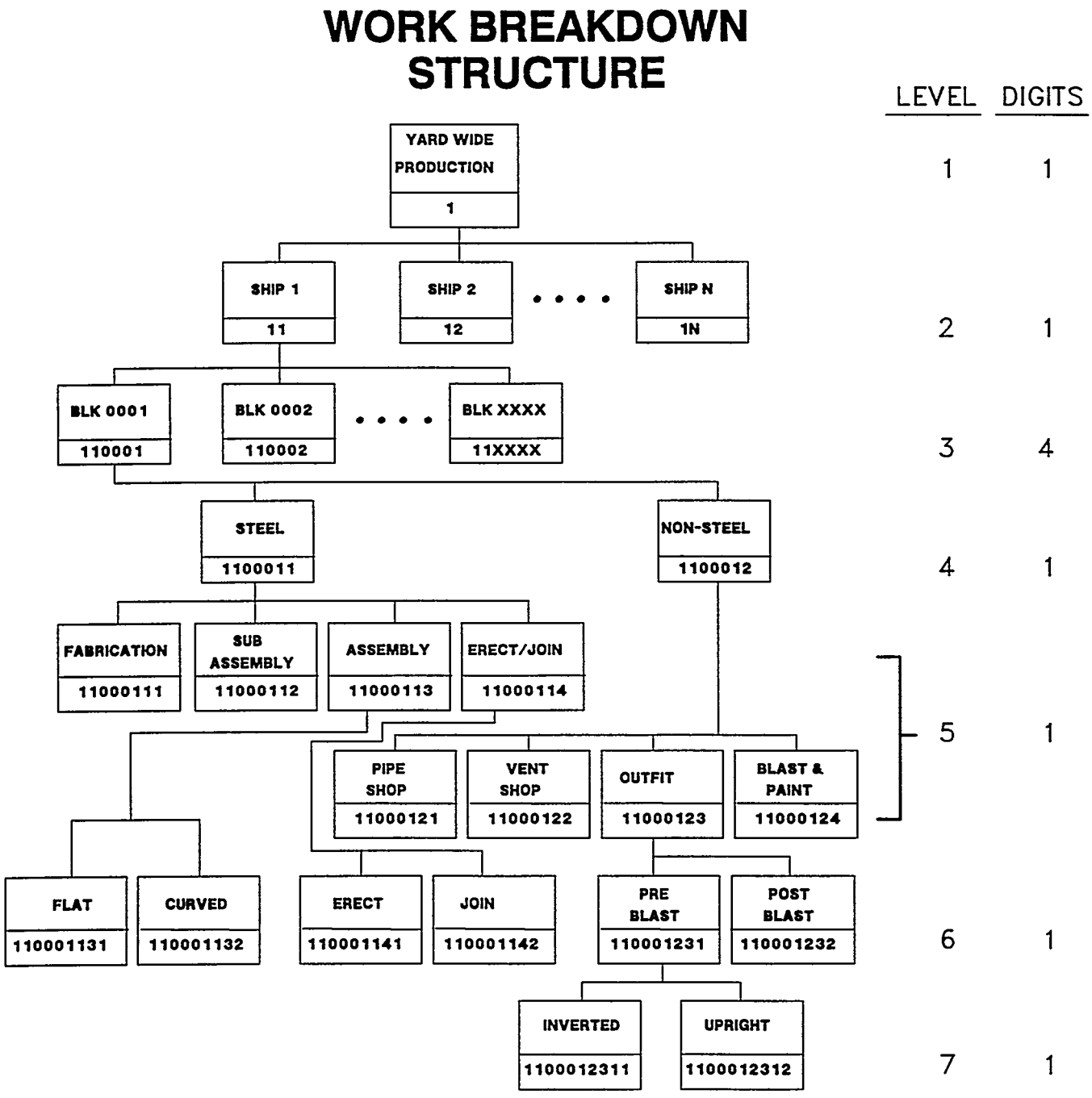


Figure 8: WBS for the MV WELL PLANNED sample.

ORGANIZATIONAL BREAKDOWN STRUCTURE

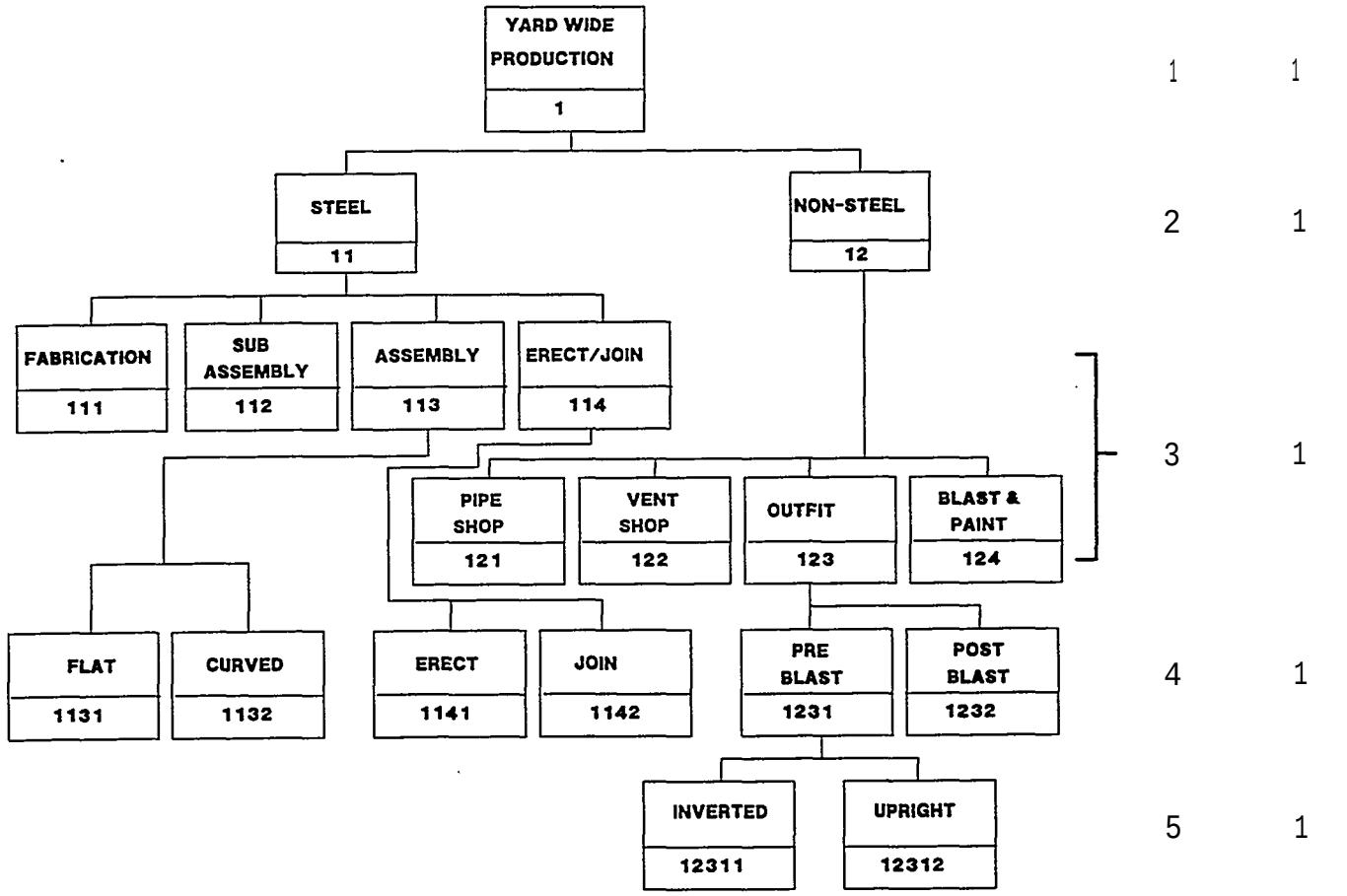


Figure 9: OBS for the MV WELL PLANNED sample.

In addition to build strategy, WBS, and OBS it is also necessary to gather information regarding resource requirements. Manning, critical materials, crane requirements, and space utilization are just a few items that can be defined as resources. Schedule refinement is an attempt to balance utilization of resources while maintaining the build strategy and meeting your project objectives. Therefore, selecting which resources to track is a key decision in the development of an Integrated Production Planning System. Resources to be tracked by the system should be those considered critical and which the yard wishes to control as Master Production Schedules are developed. The decision of which resources to track and how to code these resources should be made in the initial stages of the IPPS development. The decision will

determine the contents of the Resource Library File, the structure of the Curve Generation File, and the coding of the Model Builder and Curve Generation Programs (these files and programs will be explained later in this manual) . The resources tracked by the IPPS are shown in Table 2.

RESOURCE	RESOURCE CODE	UNIT OF MEASURE
STEELFABRICATION HOURS	H111	HOURS
STEEL SUB-ASSEMBLY HOURS	H112	HOURS
STEEL ASSEMBLY HOURS	H113	HOURS
BLAST & PAINT HOURS	H124	HOURS
OUTFlTING HOURS	H123	HOURS
ERECTION HOURS	H114	HOURS
FLAT ASSEMBLY POSITION	P1131	EACH
CURVED ASSEMBLY POSITION	P1132	EACH
BLAST & PAINT POSlTION	P124	EACH
ON-BLOCK INVERTED O/F POSITION	P12311	EACH
ON-BLOCK UPRIGHT O/F POSITION	P12312	EACH
ON-BLOCK POST BLAST O/F POSITION	P1232	EACH
ASSEMBLY COMPLETE	P-A	EACH
BLOCK ERECTION/STACKINQ	P-E	EACH

Table 2: Critical resources tracked for the MV WELL PlANNED.

Project Creation

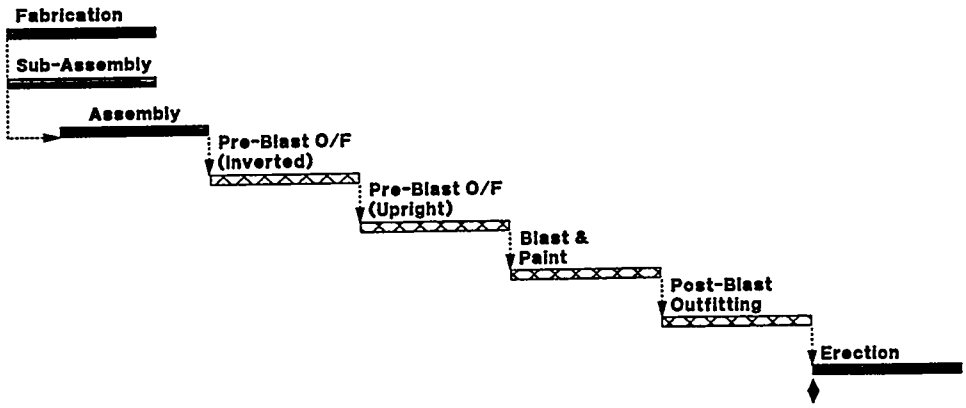
This section will describe the required format in which the production strategy information is entered for use by the IPPS. All information encompassed by the IPPS is written to and calculated from dBase files. The files used by the system and a brief explanation of how these files and structures are developed are shown in table 3. A full listing of all file structures and their required format is found in appendix I. The IPPS uses custom programs written for this project and the Open Plan software system. This User's Manual attempts to explain those items not covered in the Open Plan User's Manual. In developing and using an IPPS based upon this report, the user should refer to both this manual and that provided by Open Plan.

FILE	DESCRIPTION	dBASE STRUCTURE CREATION	dBASE RECORD CREATION
BUILD DATA	Used by model bulldor program to create the rocerds of the activity, resource, and relationship file.	Created by system user.	Created by system user from strategy sheets.
PROCESS LANE DATA	Used by model builder program to establish block to block process Into relationships.	Created by system user.	Created by system user from strategy sheets.
WBS CODE	Defines the work breakdown structure to be used to sort and group information In moaningful ways.	structure defined *Interactivoly* through Open Plan.	Created by model bullder *program.*
OBS CODE	Defines the organizational breakdown structure to be used to sort and group Information in meaningful ways.	Structure defined interactively through Open Plan.	Created by user Interation through Opon Plan.
GRAND BLOCK CODE	Defines the erectable unit to which each block Will go.	Structure defined interactively through Open Plan.	Created by model bulider program.
CALENDER	Defines work periode, *how many* dsys per week worked.	Created by Opon Plan	Developed interactively through Open Plan menu system.
HOLIDAY	Links with calendor file to define days off.	Created by Open Plan	Developed Interactively through Open Plan menu system.
RESOURCE LIBRARY	Defines rosources to *be* tracked by the system.	Created by Open Plan	Developed Interactively through Open Plan menu system.
ACTIVITY	Defines the production activities to be modeloe by the system.	Created by Open Plan, requires some modification from standard structure.	Created by model builder based upon info. contained In the Model Data Fllo. Dates determined through *Open* Plan processing.
RESOURCE	Defines the rosource requirements for each activity	Created by Open Plan, requires somes modification from standard structurs.	Created by model bullder based upon info. oontained In the Model Data Fllo.
RELATIONSHIP	Defines the relationship between the production activities.	Created by *Open* Plan, requires some modification from standard structure.	Created by modeol bullder based upon info. contained In the Model Data Fllo.
AGGREGATION	Defines the rosource requirements aggregated as a function of time.	Created by Open Plan	Generated through Open Plan processsing.
CURVE GENERATION	Rewrites the resource aggregation file In a form compatible with Freelance for curve generation.	Created by *system user.*	Generated by the *Curvo* Generation Program using the Aggregation Fllo.

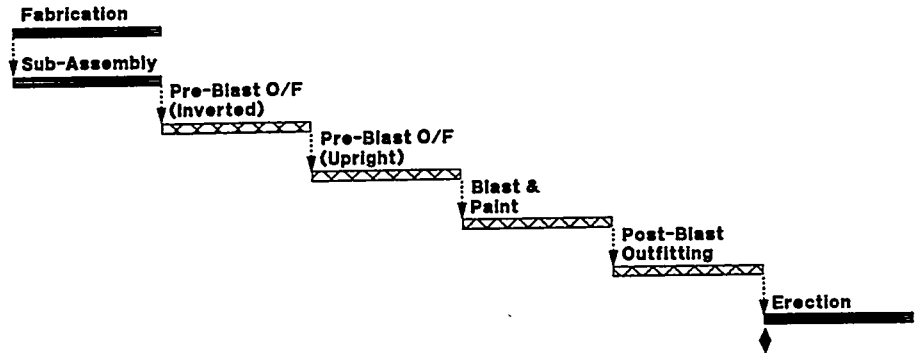
Table 3: Files required for use by the Integrated Production Planning System.

The Integrated Production Planning System models the activities associated with the assembly, outfitting, joining, and erection of hull blocks. A scheduling model with sufficient detail to meet the system objectives will be large. A model created for a 700 foot container vessel consists of approximately 3000 activities, 3000 relationships, and 10,000 resource requirements. These numbers will vary depending upon the size and complexity of the vessel and the sizes of the interim products. A yardwide integrated planning model is too large to make practical the entering of all relevant data by keyboard. Therefore, a program has been developed to build a standard setback model for a ship. This Model Builder Program uses generic strategies by block type and block specific data. Each build strategy establishes a set of activities and sequence by which a block is assembled, joined, outfitted and erected. Build strategies must be established that describes the manner in which all blocks to be tracked by the IPPS will be constructed. These build strategies are hard coded into the Model Builder Program. The build strategies used for the MV WELL PLANNED hard coded into the Model Builder Program are shown in figure 10. The Model Builder Program obtains the block specific information from the Build Data File. In addition to the Build Data File, a Process Lane Data File is created. This file establishes the link between a process lane activity and the predecessor block's process lane activity. All information entered into these files can be found on the strategy sheets. The information required by the Build Data File and Process Lane Data File, along with the strategy sheets from which this information is obtained, is shown in tables 4 and 5. The Model Builder Program creates activity, resource, and relationship files in proper format to be used by the project management software. For the detailed explanation of how the program performs this task, see the program logic flowchart in appendix IV.

Standard Build Strategy (STD)



Shop Build Strategy (SHOP)



Post Blast Stack Build Strategy (GRAND 1)

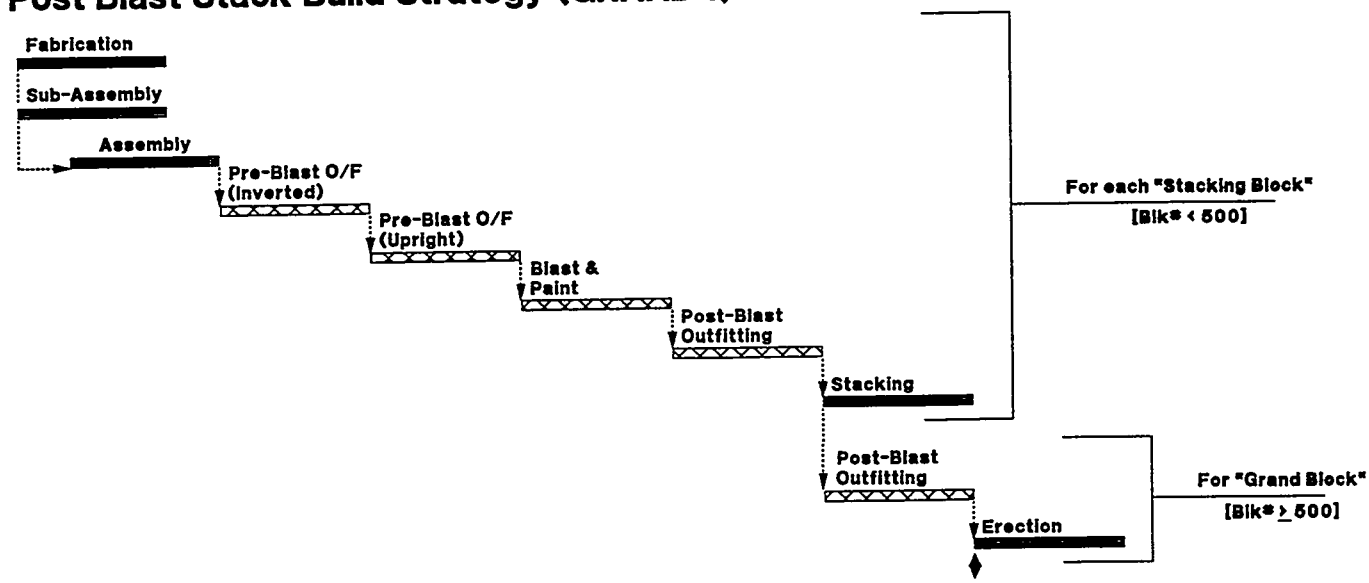
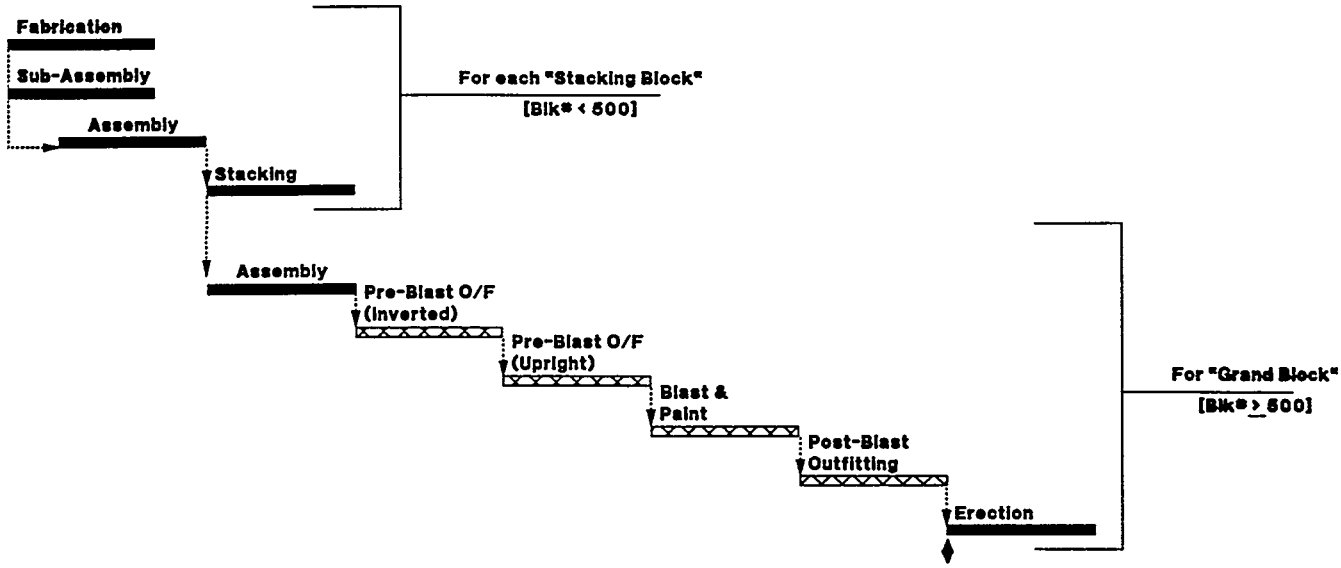
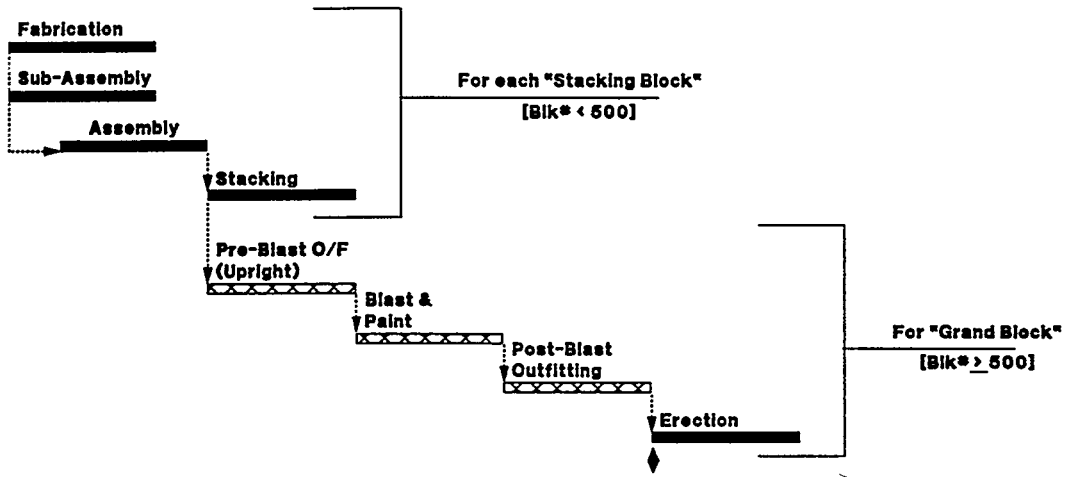


Figure 10: Strategies hard coded into the Model Builder Program for the MV WELL PLANNED.

"Assembly Stack" Build Strategy (GRAND2)



"Pre-Blast (Upright) Stack" Build Strategy (GRAND3)



"Pre-Blast (Inverted) Stack" Build Strategy (GRAND4)

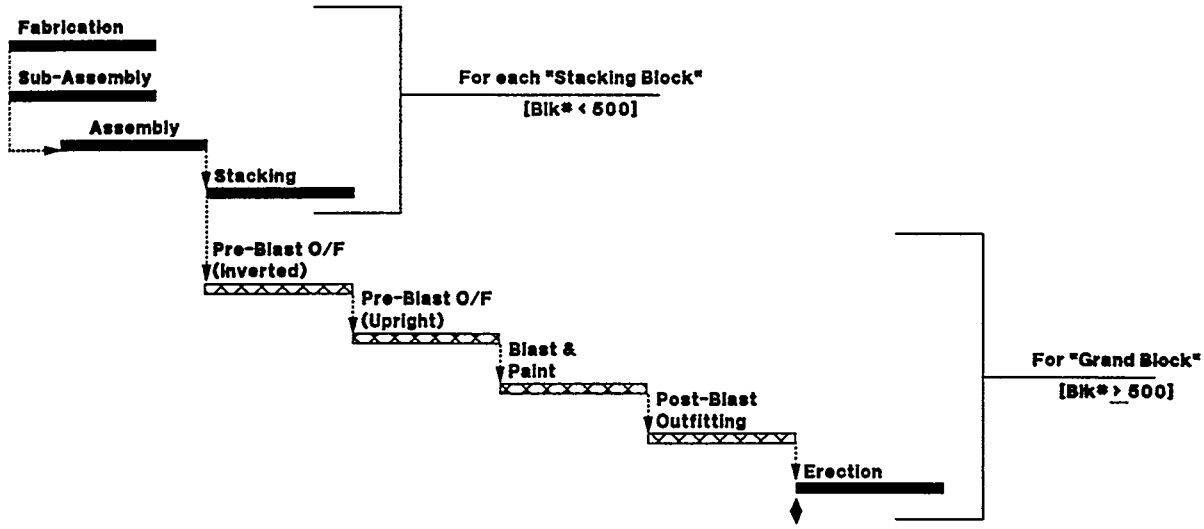


Figure 10: (Continued) Strategies hard coded into the Model Builder Program for the MV WELL PLANNED.

FIELD NAME	DESCRIPTION	INPUT FROM
DESCRIPTION	Block number	Integrated Assembly/Outfit Strat Shoot
BLK_TYP	Block typo	integrated Assembly/Outfit Strat Shoot
STRATEGY	Blook build strategy	integrated Assembly/Outfit Strat Shoot
GRAND	Grand block to which each block stacks	Grand Block Strategy Shoot
ERECTION	Data each block erects to ship (if block stacks to grand block, field will be blank.)	Erection 'Star' Chart
LAG_E	Duration from the day the first stacking block joins the grandblock to the day this block joins the grandblock.	Grand Block Strategy Sheet
ASSY_POS	Assembly build position requirement, fiat or curved build platten, to b. determined from block type.	Integrated Assembly/Outfit Strat Shoot
ASSY-D	Assembly duration	Integrated Assembly/Outfit Strat Shoot
OF1_D	Pro-blast inverted outfit duration	Integrated Assembly/Outfit Strat Shoot
OF2_D	Pre-blast upright outfit duration	integrated Assembly/Outfit Strat Sheet
OF3-D	Blast & paint duration	Integrated Aseembly/Outfit Strat Shoot
OF4_D	Post-blast outfit duration	integrated Assembly/Outfit Strat Shoot
BUD_OF1	Pro-blast Inverted outfit budget	Integrated Assembly/Outfit Strat Shoot
BUD_OF2	Pro-blast upright outfit budget	integrated Assembly/Outfit Strat Shoot
BUD_OF3	Blest & paint budget	Integrated Assembly/Outfit Strat Shoot
BUD_OF4	Post-blast outfit budget	Integrated Assembly/Outfit Strat Shoot
BUD_FAB	Fabrication budget	integrated Assembly/Outfit Strat Shoot
BUD_SUB	Sub-assembly budget	Integrated Assembly/Outfit Strat Shoot
BUD_ASSY	Assembly budget	integrated Assembly/Outfit Strat Shoot
BUD_ERECT	Erection budget	Integrated Assembly/Outfit Strat Shoot
FIRST_S	If block stocks to grand block, identifies the first block to stock to the grand block.	Grand Block Strategy Shoot

Table 4: Data fields of the Build Data File.

FIELD NAME	DESCRIPTION	INPUT FROM
ID	Activity identification for the stage/sub-stage of the block going through the process lane.	Process Lane Strategy Shoot
PRED	Activity identification for the block proceeding the block listed In ID through the process lane.	Process Lane Strategy Shoot
TYPE	Typo of relationship between activities. Since only on. block may be in a dedicated work position at a time, a finish to start (FS) relationship Will be used.	
LAG	The time interval between the activity and its prodecessor. if you want one activity to start the day after the other is oompleted, the lag is 0.	

Table 5: Data fields of the Process Lane Data File.

Note: Appendix ii contains a set of strategy sheets identifying specifically where the data for each field of the Build Data File and **Process Lane Data File is located.**

Once the Build Data File and the Process Lane Data File have been created, the Model Builder Program can be run. The program creates a series of activities, resources, and relationships for each block from the build strategies and the Build Data File. Inter-block process lane relationships are established from information in the Process Lane Data File. ID's for each activity created by the Model Builder Program are established based upon the contract letter, block number, and the stage of operation for each block . The ID coding system used by the Model Builder Program is shown in figure 11.

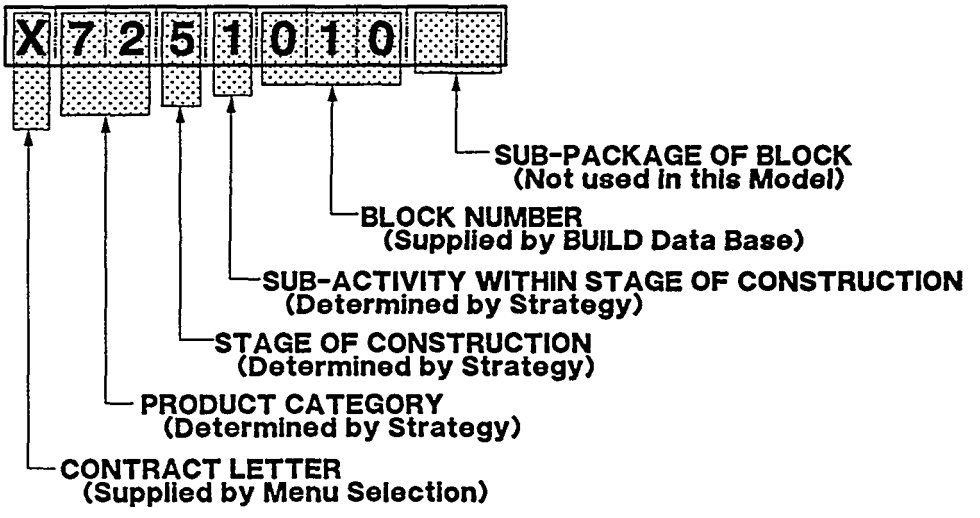


Figure 11: Activity ID as created by the Model Builder Program.

Figure 12 illustrates the activities, resources, relationships, and codes created by the Model Builder Program for a block of standard build strategy. Table 6 shows the sequence of steps necessary to build a setback model for a vessel using the IPPS.

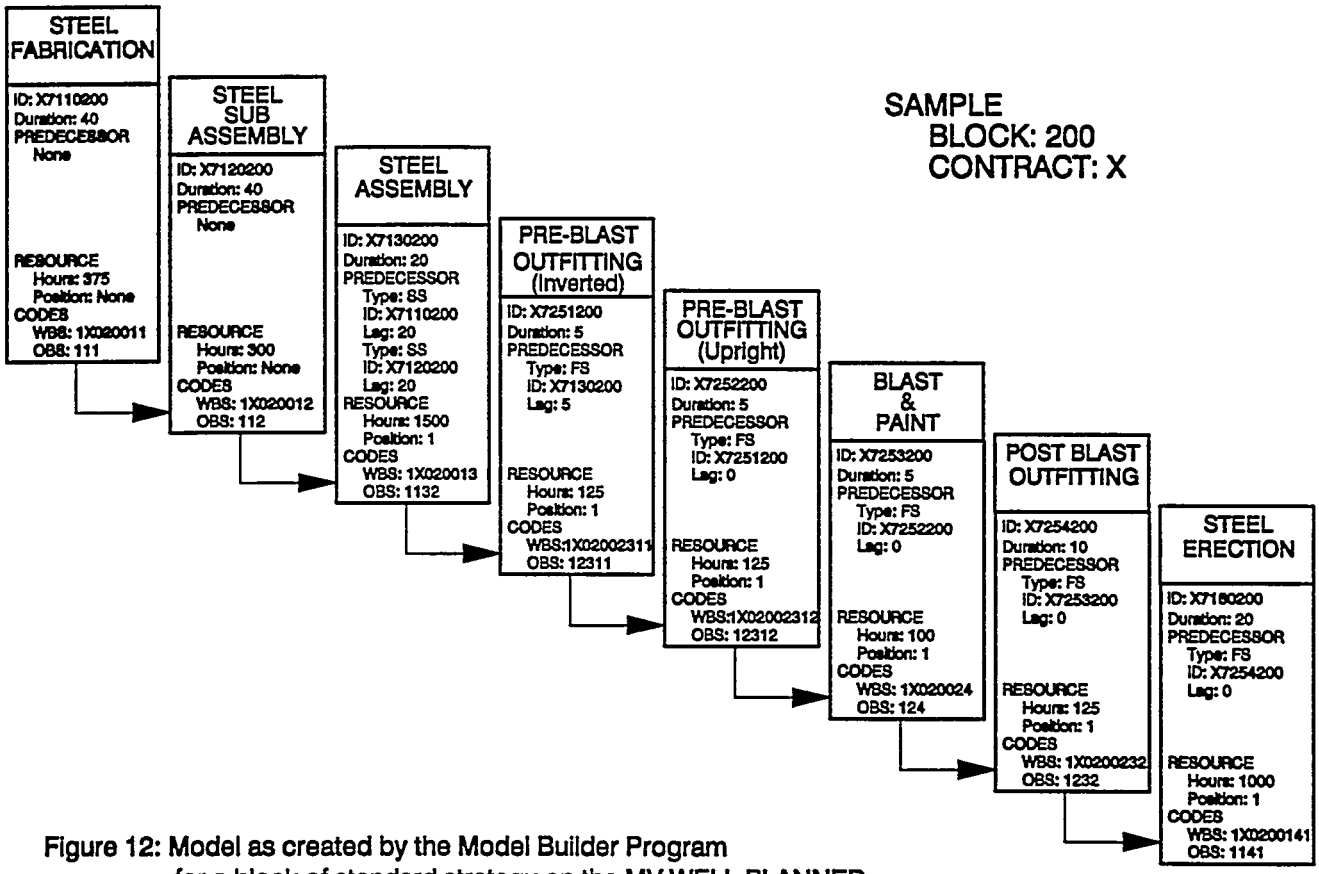


Figure 12: Model as created by the Model Builder Program
for a block of standard strategy on the MV WELL PLANNED.

(1) Create an empty Open Plan project as explained in the Open Plan manuals (e.g. WELLPLAN).

(2) Create the structures for the code files through Open Plan.
Code File 1 (Cl)- WBS
Code File 2 (C2)- OBS
Code File 4 (C4)- Grand Block Code

(3) Create the Calendar File and Holiday File through Open Plan.

(4) Build the Resource Library and the Resource Availability files (e.g. OVERALL) through Open Plan showing the resources the system is to track as in table 2. Create the structure of the Curve Generation File (e.g. CURVES) as shown in table 6A.

(5) Build the OBS file (Code C2) through Open Plan. A sample of the OBS file created for the MV WELL PLANNED is shown in Appendix III.

Note: A common Calendar File, Holiday File, OBS File, and Resource Library File should be used for all projects being tracked by the IPPS.

(6) Generate a setback model through the Model Builder Program.
- Select the Integrated Plan System menu choice under the Process heading of Open Plan's main menu.
  - Select menu option 1: New Model Generation.
  - Select menu option 1: Model Builder Program.
    - Enter the contract letter designator of the ship for which you are building the model (you must have a separate contract letter for each vessel being modeled by the IPPS).
    - Enter the project name corresponding to the project name created in step (1) along with the path to which you want the data files created by the IPPS written. This path is the path to which Open Plan writes each of its projects (e.g. c:\WELLPLAN).
    - Enter the name of the Build Data Base with path (e.g. c:\BUILD).

  The Model Builder Program creates the activity, resource, and relationship file in Open Plan for each block and grand block. If the Model Builder Program runs successfully, it returns a message saying the project has been created and asks if you wish to add any process lane relationships.
  - Enter the field name and path of the Process Lane Data File (e.g c:\P_LANE).

  This adds the block to block process lane relationships. The system then asks if you wish to build the WBS code file (C1).
  - Enter 'Y'.
  - Enter the WBS code file path and *name* as created in step (2) (e.g. c:\sP8_cl).

  This creates the Cl data file. The system then asks if you wish to create a new grand block code field (C4).
  - Enter 'Y'.
  - Enter the Grand Block Code File path and name as created in step (2) (e.g. c:\SP8_C4).
- Escape back to the Open Plan main menu.

(7) Reindex the model by selecting the Rebuild Indexes menu choice under the System menu heading of Open Plan's main menu. When in the Rebuild Index facility of Open Plan select 1 and 2 and then 9 to exit.

Table 6: Model creation sequence of steps.

FIELD NAME	*DESCRIPTION*
TODATE	LAST DAY OF EACH AGGREGATION PERIOD
WORKPDS	NO. OF WORK DAYS SCHEDULED IN AGG. PERIOD
H111	STEEL FABRICATION HOURS
H112	STEEL SUB-ASSEMBLY HOURS
H113	STEEL ASSEMBLY HOURS
H1141	STEEL ERECT HOURS
H1142	STEEL JOIN HOURS
H114	STEEL JOIN/ERECT HOURS
H12311	PRE-BLAST INVERTED OUTFIT HOURS
H12312	PRE-BLAST UPRIGHT OUTFIT HOURS
H1231	PRE-BLAST OUTFIT HOURS
H1232	POST-BLASTOUTFIT HOURS
H123	TOTALOUTFITHOURS
H124	BLAST&PAINTHOURS
H11	TOALSTEELHOURS
H12	TOTALOUTFIT/BLAST &PAINTHOURS
H1	TOTALHOURS
P_A	BLOCK ASSEMBLIES COMPLETE
P_E	BLOCK ERECTIONS COMPLETE
P1131	FLAT ASSEMBLY POSITIONS IN USE
P1132	CURVED ASSEMBLY POSITIONS IN USE
P113	TOTAL ASSEMBLY POSITIONS IN USE
P12311	PRE-BLAST INV. OUTFIT POSITIONS IN USE
P12312	PRE-BLAST UPRIGHT OUTFIT POSITIONS IN USE
P1231	PRE-BLAST OUTFIT POSITIONS IN USE
P1232	POST-BLAST OUTFIT POSITIONS IN USE
P123	TOTAL OUTFIT POSITIONS IN USE
P124	BLAST & PAINT POSITIONS IN USE
P12	TOTAL O/F AND B & P POSITIONS IN USE

Table 6A Structure of the Curve Generation File for the MV WELL PLANNED.

Schedule Development

The steps described in table 6 will create a model for a single hull . However, the IPPS tracks the ground assembly, outfit, joining, and erection of all projects in process and planned for the yard. It is therefore necessary to merge the model created by the Model Builder Program (e.g. WELLPLAN) with a single, yardwide Open Plan project (e.g. OVERALL). This model contains all the work to be tracked by the IPPS on current and planned contracts. The project OVERALL has the same Calendar File, Holiday File, OBS file, and Resource Library File as that of the individual vessel models. The WBS and Grand Block Code Fles for OVERALL are a combination of the WBS and Grand Block Code Files of the individual vessels being tracked. To merge the project WELLPLAN into the project OVERALL, follow the sequence of steps shown in table 7.

(1)	Select the Merge Copy menu choice under the Utilities heading of Open Plan's main menu.
(2)	Choose menu option 2: Merge l Enter the project to contain the merged projects (e.g. OVERALL) . l The program asks if you want to delete all project data. Enter "No". l Enter the project you are merging into the model (e.g. WELLPLAN) . l The program asks if you need to renumber. Enter 'No'. l The program asks if you wish to merge another project. Leave blank, hit return.
(3)	Enter '9' *to* return to the Open Plan main menu.

Table 7: Model merge sequence of steps.

Once the projects have been combined into an overall model, time analysis may be run to determine the first cut schedule of activities tracked by the IPPS. Before time analysis is run, the only dates in the system are the fixed dates on which blocks or grand blocks erect to the ship. Time analysis will create three sets of dates for each activity: early, late, and scheduled. The model is constrained only by the back end erection activities.

Since there is no constraint placed on early dates, the only dates that have any meaning to the system are the late dates. All programs within the IPPS use late dates. The sequence of steps to perform time analysis is explained in the Open Plan manual. The processed yardwide model must be analyzed to see if the manning and facility utilization implied by the schedule are acceptable. Manning and facilities are investigated by creating resource utilization curves and laydown charts. Resource utilization data is generated by creating the Aggregation File for the overall model. The Aggregation File is then processed by the Data Program of the IPPS. The dBase file created by the Data Program is imported into a graphics software package and resource utilization curves are produced. Table 8 shows the sequence of steps necessary to create the resource utilization curves. For the detailed explanation of how the Build Program performs its task, see the program logic flowchart in appendix IV. A sample of the resource utilization curve created by the IPPS is shown in figure 13.

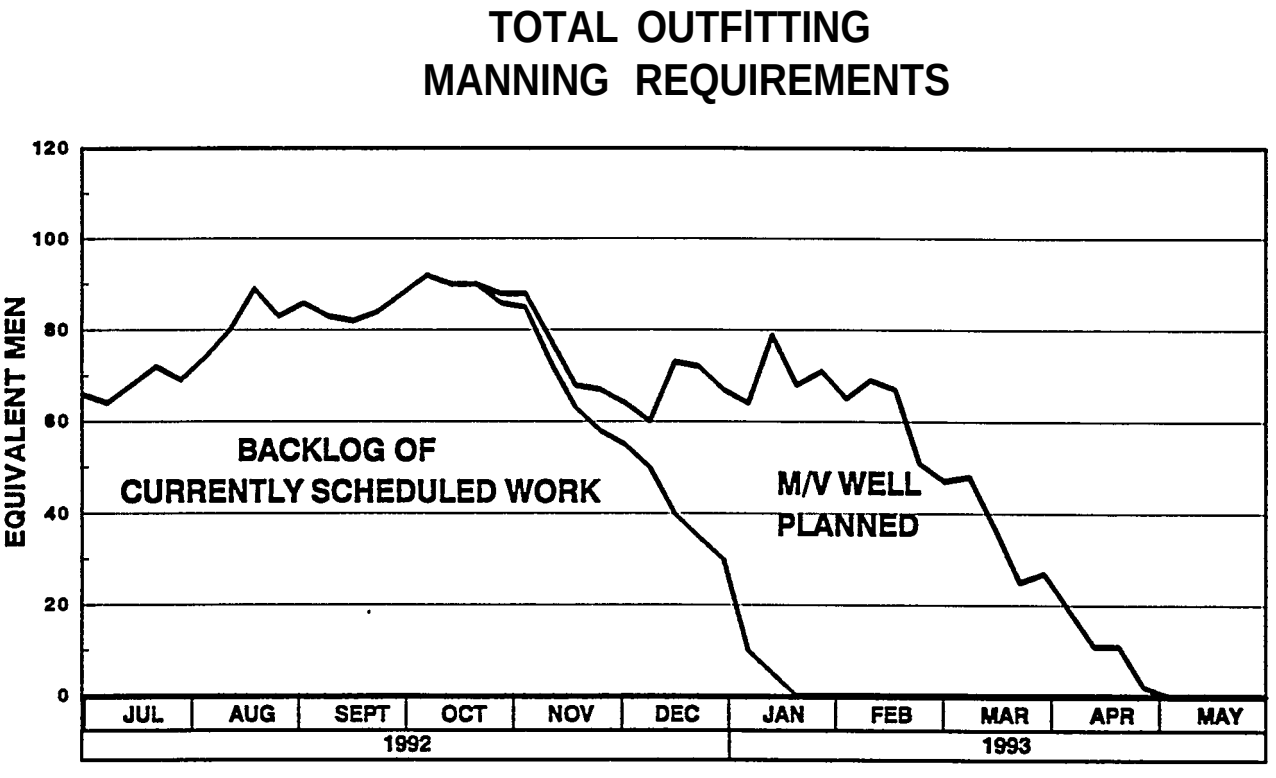


Figure 13: Resource utilzation curve developed through the IPPS for total outfitting manning requirements (resource code H123.)

(1) Select the General Reports menu choice under the Reports heading of Open Plan's main menu.

- Select F5 for Resource.
- Select HISTRES- Resource histogram from aggregation file.
- Enter the model name (e.g. OVERALL) of project for which you are creating the resource utilization curves.
- The system asks if you need to do aggregation:'Y'.
- Enter the resource availability file (e.g. OVERALL). This file is the resource availability file created in step (4) of table 6.
- Enter the aggregation file name (e.g. OVERALL).
- Enter the aggregation period length (e.g. 7 for weekly aggregation).
- Enter the aggregation start date (e.g. 12/30/91). It is best to select a Monday so that the ending date of the aggregation period will include a full work week and end *on* Sunday.
- Enter number of aggregation periods (e.g. 104 for 2 Years). You can select whatever period you feel appropriate, the time span should be at least as long as the span of the project for which you are developing schedules.

Open Plan will create the aggregation file (OVERALL.agg in this example) containing data regarding the usage over time of each resource defined in the resource library.

- When the aggregation file is complete, the system returns asking for resource codes for reporting purposes. At this point, escape from the system by entering '*'.
- Enter '*' again to return to Open Plan's main menu.

(2) Select the Integrated Plan System menu choice under the Process heading of Open Plan's main menu.

- Enter '2': Project Analysis.
- Enter 'l': Data Generation Program.
- Enter the aggregation file name and path as created in step (1) above (e.g. C:\OVERALL).
- Enter the curve generation file path and name as created in step (4) of table 6 (e.g. C:\CURVES).
- Enter the number of days per period, start date, and number of aggregation periods as was entered when creating the aggregation file in step (1) of this procedure.

The data generation program will take the Aggregation File developed by Open Plan and rewrite it to the Curves Data File in a form suitable for curve generation by the graphics software.

- Escape back to Open Plan main menu.

(3) Import the fields you are investigating (e.g. Hll if you are investigating total steel hour utilization) into a graphics software program and print out the curves. It is recommended that standard templates be developed within the graphics program for creation of resource utilization curves. The start date and time span shown on the templates should correspond to the aggregation and curves file generated in the previous steps.

Table 8: Resource utilization curve generation sequence of steps.

After resource utilization curves have been analyzed and preliminary schedule leveling has been accomplished, laydown charts are developed. These charts graphically display space utilization within a yard by showing the location of blocks as a function of time. The Laydown Generation Program uses the activity file of the Overall Production Model after processing by the project management software. The program takes the activity file and extracts the dates and locations necessary to develop laydown schedules for each production area. For a detailed explanation of how the Laydown Generation Program performs its task, see the program logic flowchart in appendix IV. The IPPS as currently developed has no automated method for assigning laydown positions to activities. The assignment is made by manually inputting the laydown locations into the "TABLE" field of the Overall Model activity file. Automation of the laydown position assignment process is an area which the system users may wish to explore. Once all activities for which laydown charts are developed have been assigned a laydown location, the laydown charts may be produced. The sequence of steps required to produce the laydown charts are shown in table 9. A sample of a laydown chart created by the IPPS is shown in figure 14.

(1) Select the Integrated Plan System menu choice under the Process heading of Open Plan's main menu.
(2) Select 4: Production Schedule Update.
(3) Select 3: Laydown Schedules.
(4) Enter the model name with path (e.g. C:\OVERALL)'.
The program will set up required fields and re-index.
(5) The program will ask if you wish to recalculate placements. If laydown locations or schedule changes have been made since the laydown was last processed, enter 'Y'. If no changes have been made since the last laydown processing, enter 'N' and the program advances to the next step without recalculating.
(6) Place an 'X' by all areas for which you wish laydown tables printed. Place an 'X' by all time periods for which you wish laydowns printed.
The program will produce the laydown charts requested.
(7) Escape back to the Open Plan main menu.

Table 9: Laydown chart generation sequence of steps.

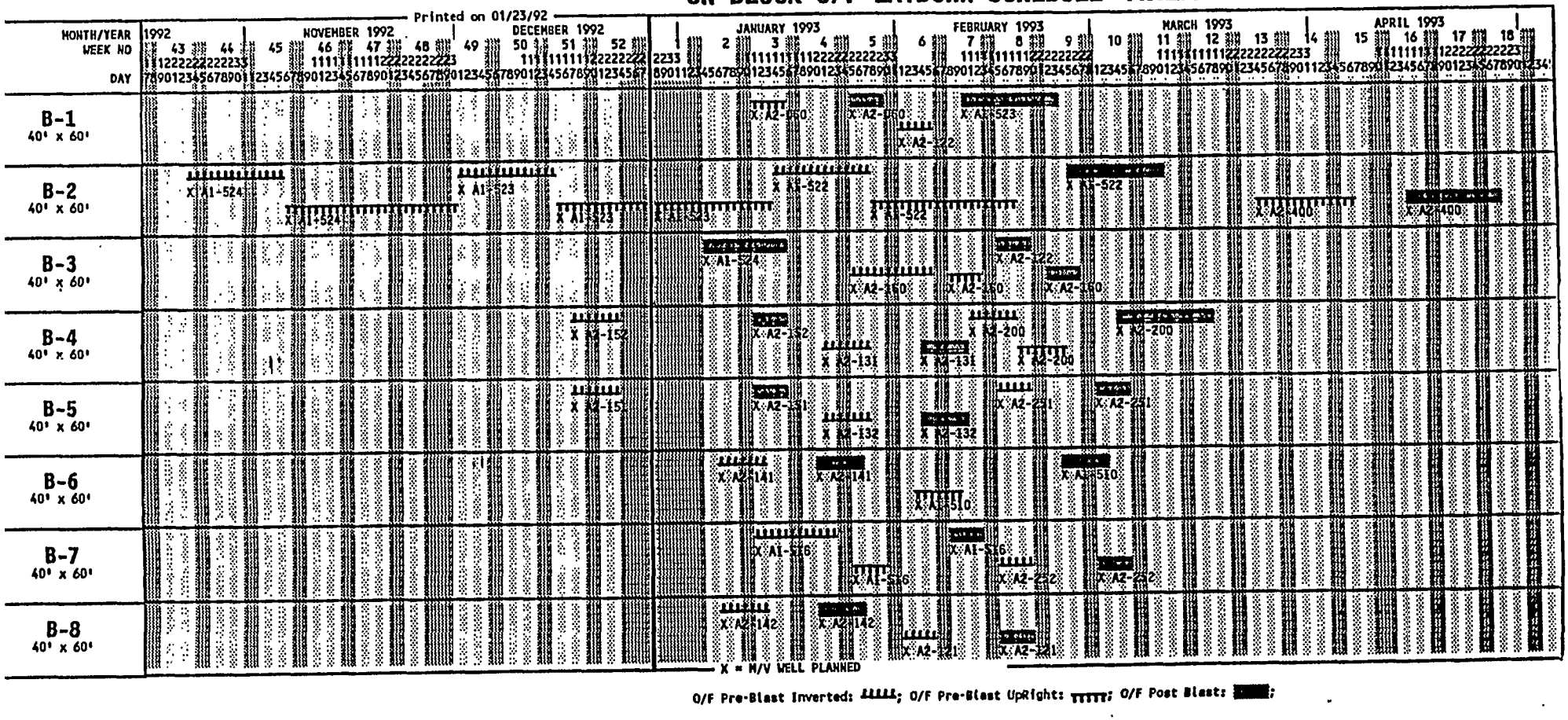


Figure 14: Laydown charts produced by the Integrated Production Planning System showing the scheduled laydown of blocks in outfitting area 'B' for the M/V WELL PLANNED.

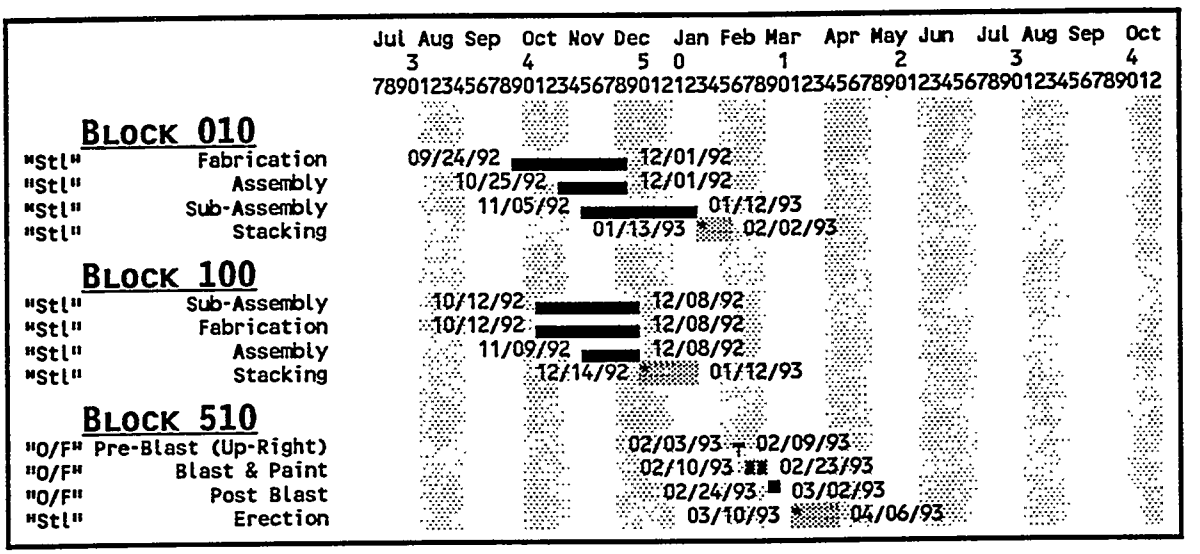
Once scheduling iterations have been performed and a credible schedule developed, the IPPS will produce Gantt charts for each of the blocks and grand blocks. Table 10 shows the sequence of steps necessary to create the Gantt charts. For a detailed explanation of how the Gantt Generation Program performs its task, see the program logic flowchart in appendix IV.

(1)	Select the Integrated Plan System menu choice under the Process heading of Open Plan's main menu.
(2)	Select 3: Baseline Schedule Generation.
(3)	Select 2: Master Schedules (Gantt Chart Form).
(4)	Enter the model name with path of project for which you wish to generate Gantt charts (e.g. C:\OVERALL).
(5)	Enter the contract letter of the block for which you wish to view the Gantt chart.
(6)	Enter the block number for which you wish to view a Gantt chart. The Gantt chart and table of activity dates for the block requested will appear on the screen. The program asks if you wish a hard copy.
(7)	To print a hard copy of the Gantt chart enter 'Y'.
(8)	Escape back to Open Plan main menu.

Table 10: Gantt chart generation sequence of steps.

When a Gantt chart for a grand block is requested, only the activities at the grand block level appear on the screen. When a hard copy of the Gantt chart is printed, both the activities for the grand block and the blocks stacking to the grand block are shown . A sample of a grand block Gantt chart produced by the IPPS is shown in figure 15.

GRAND BLOCK 510 BUILD STRATEGY



GRAND BLOCK 510 BUILD STRATEGY

BLOCK NUMBER : 010

STEEL	Start	Comp	Dur	OUTFITTING	Start	Comp	Du
Fab(711)	09/24/92	12/01/92	40	O/F Inv.(7251)			
Sub-Assy(712)	11/05/92	01/12/93	40	O/F Up-R(7252)			
Assembly(713)	10/25/92	12/01/92	20	S/B & P(7253)			
Stacking(715)	01/13/93	02/02/93	15	O/F Post(7254)			

BLOCK NUMBER : 100

STEEL	Start	Comp	Dur	OUTFITTING	Start	Comp	Du
Fab(711)	10/12/92	12/08/92	40	O/F Inv.(7251)			
Sub-Assy(712)	10/12/92	12/08/92	40	O/F Up-R(7252)			
Assembly(713)	11/09/92	12/08/92	20	S/B & P(7253)			
Stacking(715)	12/14/92	01/12/93	15	O/F Post(7254)			

BLOCK NUMBER : 510

STEEL	Start	Comp	Dur	OUTFITTING	Start	Comp	Du
Fab(711)				O/F Inv.(7251)			
Sub-Assy(712)				O/F Up-R(7252)	02/03/93	02/09/93	5
Assembly(713)				S/B & P(7253)	02/10/93	02/23/93	10
Stacking(715)				O/F Post(7254)	02/24/93	03/02/93	5

Figure 15: Gantt charts produced by the Integrated Production Planning System for Grand Block 510 of the MV WELL PLANNED.

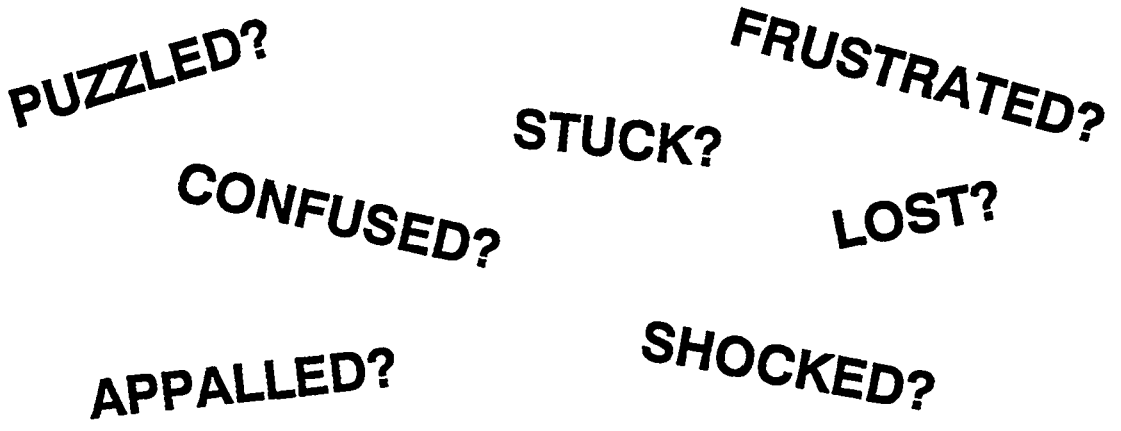
Progressing and Updating the Model

A schedule is a dynamic rather then a static document. For schedules to remain current and meaningful, they must be updated based upon input from all affected parties. This input may be received at regularly scheduled meetings where production plans are updated to reflect actual and projected progress. These meetings are described in the Project Report section of this development. The IPPS model may be updated and progressed as explained in the Open Plan User's manual. Revised schedules, manning curves, and laydowns are produced from the updated model.

CUSTOMIZING THE SYSTEM FOR USE IN YOUR YARD

As stated earlier, the Integrated Production Planning System presented in this report is not intended to be a turn key system. To apply the methods and techniques of this report, a yard must establish a team to develop a planning system. The IPPS should not be viewed as a computer system, it is a production planning system that makes use of computer tools. Simply obtaining and installing the software will not give a shipyard an operable system. To be successful, a yard must expend the time and effort necessary to develop and maintain the system. Coding must be altered to reflect the yard's facilities, production strategies, organization, and work breakdown. Decisions must be made as to which resources should be tracked by the IPPS. Developing an Integrated Production Planning System is a significant task. However, once developed, the IPPS is a valuable tool that will assist shipyard personnel in making effective production decisions.

HELP!!!!!

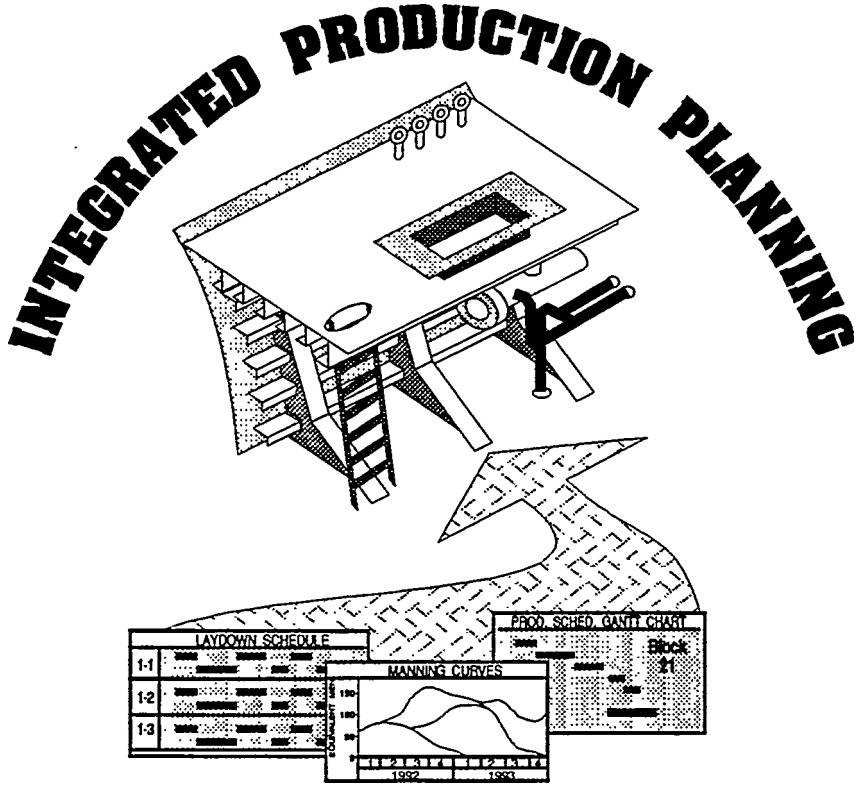


If you can track us down, we would like to help. Our number and address as of Spring '92 is shown below. Please do not hesitate to write or call.

Rich Neumann or Dave McQuaide
MS 07
National Steel & Shipbuilding Co.
Harbor Drive & 28th St.
San Diego, Ca. 92186-5278

call (61 9) 544-3583 for Rich Neumann
or (619) 544-8481 for Dave McQuaide

APPENDIX



SYSTEM

APPENDIX

I. LISTING OF THE FILE STRUCTURES USED BY THE IPPS
II. STRATEGY SHEET LOCATION OF BUILD DATA FILE INFORMATION
III . OBS CODE DATA FILE
IV. PROGRAM LOGIC FLOW CHARTS
   - Ž MENU SYSTEM
   - Ž MODEL BUILDER
   - Ž RESOURCE UTILIZATION CURVE GENERATION
   - Ž LAYDOWN CHART GENERATION
   - Ž GANTT CHART GENERATION

V. PROGRAM CODE
VI. APPLICATION OF PC BASED FACILITIES SIMULATION

APPENDIX I

LISTING OF THE FILE STRUCTURES
USED BY THE IPPS

BUILD. dbf - Build Data File

Field	Type	Width
BLOCK	Character	3
BLK_TYPE	Character	12
STRATEGY	Character	6
GRAND	Character	3
ERECTION	Date	8
LAG_E	Numeric	3
ASSY_POS	Numeric	6
OFl_D	Numeric	6
OF2_D	Numeric	6
OF3_D	Numeric	6
OF4_D	Numeric	6
BUD_OFl	Numeric	6
BuD_oF2	Numeric	6
BuD_oF3	Numeric	6
BuD_oF4	Numeric	6
BUD_FAB	Numeric	6
BUD_SUB	Numeric	6
BUD ASSY	Numeric	6
BUD_ERECT	Numeric	6
FIRST_S	Character	3

P_LANE.dbf - Process Lane Data File

Field	Type	Width
ID	Character	10
PRED	Character	10
TYPE	Character	2
GRAND	Character	3
LAG	Numeric	3
RELTF	Numeric	5
RELFF	Numeric	5
PLOTJY	Numeric	4
PLOTIX	Numeric	4
PLOTAY	Numeric	4
PLOTIY	Numeric	4
PLOTJX	Numeric	4
RELCAL	Numeric	3

WBS_C1.cod - Code File 1 (WBS)

Field	Type	Width
C1	Character	10
CIDESC	Character	60

OBS_C2 . cod - Code File 2 (OBS)

Field	Type	Width
C2	Character	10
C2DESC	Character	60

GRD C4. cod - Code File 4 (Grand Block Code)

Field	Type	Width
C4	Character	10
C4DESC	Character	60

PROJECT.cal - Calender Data File

Field	Type	Width
CALENDER	Numeric	3
HOLIDAYFIL	Character	8
DAY1	Numeric	1
DAY2	Numeric	1
DAY3	Numeric	1
DAY4	Numeric	1
DAY5	Numeric	1
DAY6	Numeric	1
DAY7	Numeric	1

PROJECT.hol - Holiday Data File

Field	Type	Width
HOLIDAY	[illegible]	8

PROJECT.rds - Resource Library Data File

Field	Type	Width	Dec
RESCODE	Character	6	
RESDESC	Character	20	
UNIT	Character	10	
UNITCOST	Numeric	10	2
THRESHOLD	Numeric	5	
RESTYPE	Character	1	
RESCLASS	Character	1	
SUPPRESS	Character	1	

MODEL.act & MODELA.act - Activity Data File for projects MODEL and MODELA

Field	Type	Width
ID	Character	10
D	Numeric	4
CAL	Numeric	3
DS	Character	30
TYPE	Character	1
PROG	Numeric	3
TARGS	Date	8
TARGF	Date	8
c1	Character	10
C2	Character	10
C3	Character	10
C4	Character	10
FF	Numeric	5
TF	Numeric	5
ESDATE	Date	8
EFDATE	Date	8
LSDATE	Date	8
LFDATE	Date	8
SSDATE	Date	8
SFDATE	Date	8
CRITICAL	Numeric	1
LOGICODE	Character	1
COMPSTAT	Numeric	1
ACTS	Date	8
ACTF	Date	8
BSTART	Date	8
BFINISH	Date	8
BCOST	Numeric	9
ACOST	Numeric	9
PCOMP	Numeric	3
ACOST_TLP	Numeric	9
PCOMP_TLP	Numeric	3
RESCOST	Numeric	9
TARGSTYPE	Character	2
TARGFTYPE	Character	2
FEDATE	Date	8
DELAYRES	Character	6
COMPREMDUR	Numeric	4
XFDATE	Date	8
ACTYPE	Character	1
MAXD	Numeric	4
MAXSPLITS	Numeric	2
MINSPLITD	Numeric	4
RSCLASS	Character	1
RESLABUNIT	Numeric	9
FTF	Numeric	5
SDUR	Numeric	4
PLACEMENT	Character	2
TABLE	Character	5
SIZE	Character	5

MODEL. res & MODELA. res - Resource Data File for projects MODEL and MODELA

Field	Type	Width
ID	Character	10
RESCODE	Character	6
LEVEL	Numeric	5
OFFSET	Numeric	3
PERIOD	Numeric	3
LEVTYPE	Character	1

MODEL. rel & MODELA.rel - Relationship Data File for projects MODEL and MODELA

Field	Type	Width
ID	Character	10
PRED	Character	10
TYPE	Character	2
LAG	Numeric	3
RELTF	Numeric	5
RELFF	Numeric	5
PLOTJY	Numeric	4
PLOTIX	Numeric	4
PLOTAY	Numeric	4
PLOTIY	Numeric	4
PLOTJX	Numeric	4
RELCAL	Numeric	3

PROJECT. aVl - Resource Availability Data File

Field	Type	Width
RESCODE	Character	6
LEVEL	Numeric	5
FROMDATE	Date	8
TODATE	Date	8
RESCAL	Numeric	3

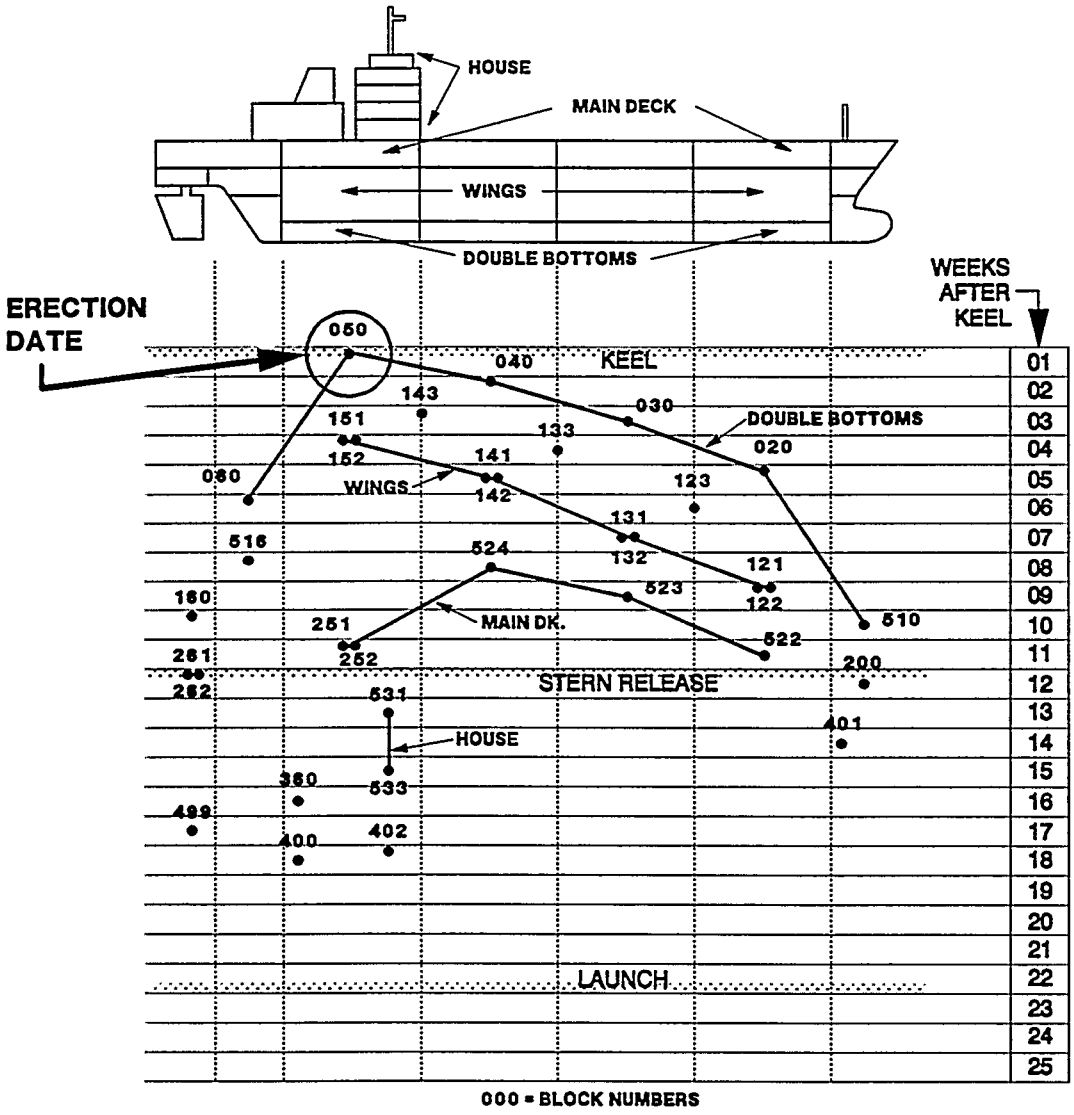
CURVE . dbf - Curve Generation Data File

Field	Type	Width
TODATE	Date	8
WORKPDS	Numeric	5
Hill	Numeric	5
H112	Numeric	5
H1131	Numeric	5
H1132	Numeric	5
H1133	Numeric	5
H113	Numeric	5
H1141	Numeric	5
H1142	Numeric	5
H114	Numeric	5
H12311	Numeric	5
H12312	Numeric	5
H1231	Numeric	5
H1232	Numeric	5
H123	Numeric	5
H124	Numeric	5
Hll	Numeric	5
H12	Numeric	5
H1	Numeric	5
P A	Numeric	5
PEE	Numeric	5
P1131	Numeric	5
P1132	Numeric	5
P1133	Numeric	5
P113	Numeric	5
P12311	Numeric	5
P12312	Numeric	5
P1231	Numeric	5
P1232	Numeric	5
P123	Numeric	5
P124	Numeric	5
P12	Numeric	5

APPENDIX II

STRATEGY SHEET LOCATION OF
BUILD DATA FILE INFORMATION

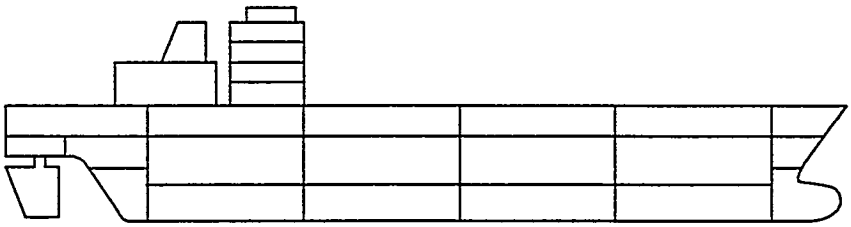
ERECTION "STAR" CHART



000 = BLOCK NUMBERS

Erection "Star" Chart showing data gathered for use by the Model Builder Program.

INTEGRATED ASSEMBLY/OUTFIT STRATEGIES BY BLOCK TYPE



BLOCK

BLK_TYPE

STRATEGY

ASSY_POS
(Based upon block type)

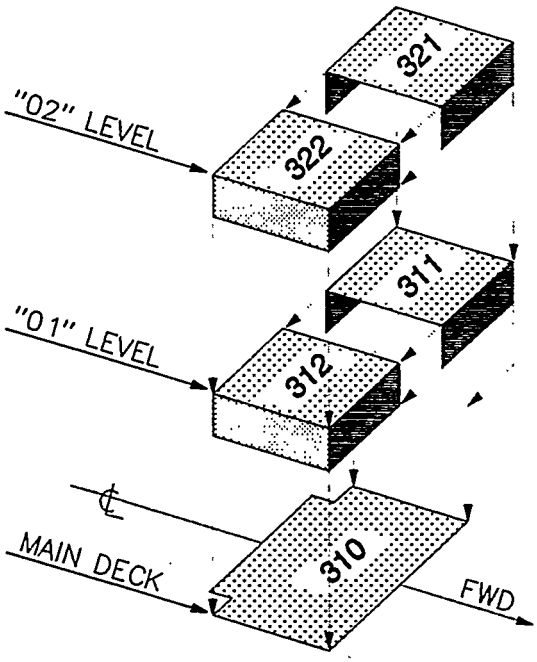
BLK TYPE	BLOCK NUMBERS	BUILD STRAT.	ASSY	PRE-BLAST INVERTED	PRE-BLAST UP RIGHT	BLAST & PAINT	POST BLAST	ERECT/ STACK
DOUBLE BOTTOM	020,030 040,050	STANDARD	20 1500	–	5 250	5 100	5 250	20 1000
TRANS. BHD.	123, 133 143	STANDARD	15 500	5 150	–	5 100	5 150	20 400
STERN	160,261 262	STANDARD	15 2000	10 300	5 150	5 100	5 150	20 1500
STERN	161, 162	GRAND2	15 1000	–	–	–	–	20 200
MAIN DECK	251,25	STANDARD	15 800	5 200	–	5 100	5 200	20 500
MAIN DECK	221–>242	GRAND4	15 1000	–	–	–	–	20 150
WING	121–> 152	STANDARD	15 800	5 150	–	5 100	5 150	20 600
BOW	200	STANDARD	20 *A* 1500 *F*	5 *B* 125 *G*	5 *C* 125 *H*	5 *D* 100 *I*	10 *E* 125 *J*	20 *K* 1000
BOW	010, 100	GRAND3	20 2000	–	–	–	–	20 350
HOUSE	310–>350	GRAND 1	15 800	20 800	5 200	5 100	–	20 300
MAST	401,402	SHOP	–	10 300	–	5 100	20 700	20 350
RUDDER	499	STANDARD	30 1000	–	–	5 100	–	20 650
CASEING	360	STANDARD	15 900	10 500	10 500	5 100	10 500	20 650
STACK	400	STANDARD	20 1000	–	10 500	5 100	10 500	20 600

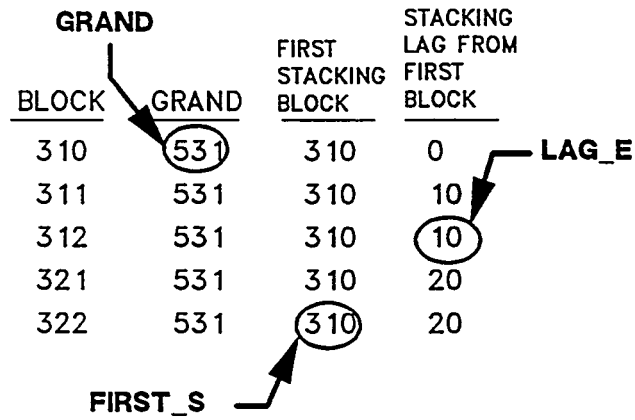
Integrated assembly/outfit strategies showing data gathered for use by the Model Builder Program.

A: ASSY_D
B: OF 1_D
C: OF2_D
D: OF3_D
E: OF4_D

F: BUD_ASSY
G: BUD_OF 1
H: BUD_OF2
I: BUD_OF3
J: BUD_OF4
K: BUD_ERECT

GRAND BLOCK STRATEGY
LOWER HOUSE - G.B. 531

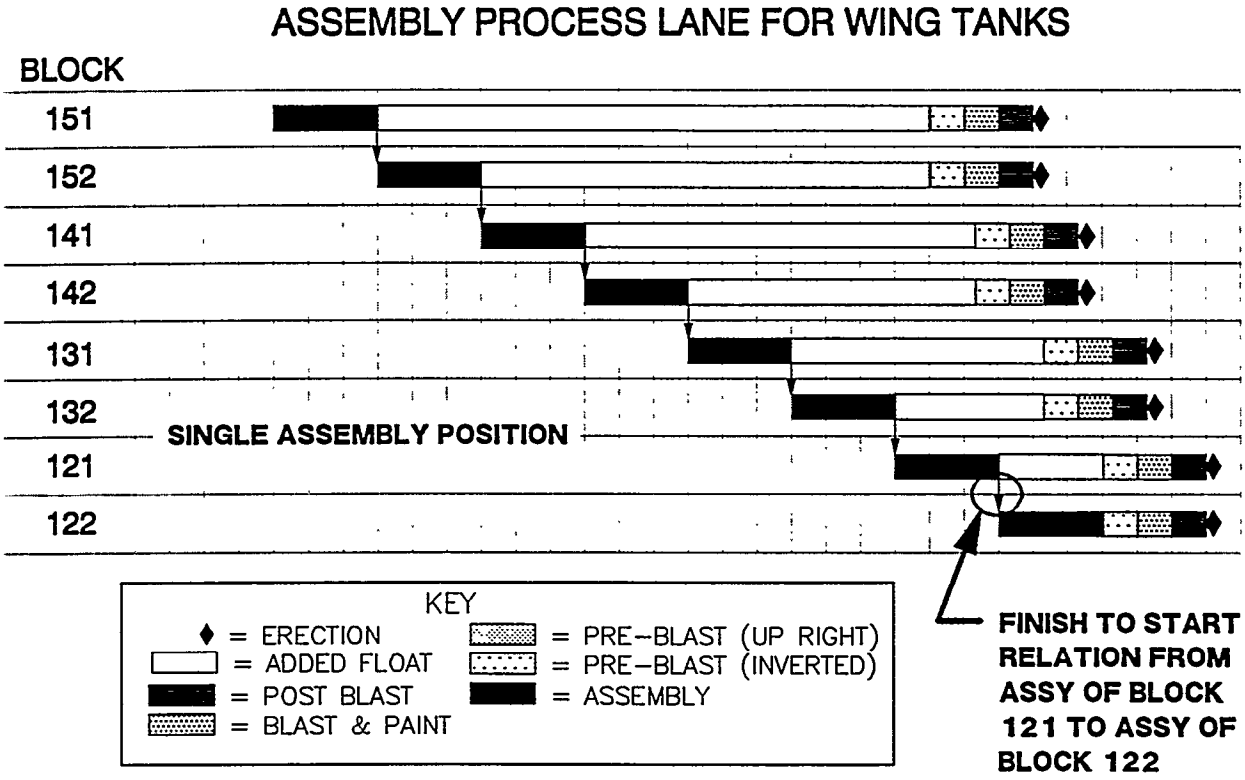




BLK TYPE	BLOCK NUMBER	GRAND	BUILD STRAT.	ASSY	PRE-BLAST INVERTED	PRE-BLAST UP RIGHT	BLAST & PAINT	POST BLAST	ERECT/ STACK
HOUSE GB	531	531	GRAND2	—	—	40 2500	5 700	15 1500	20 1200

Grand block strategy sheet showing data gathered for use by the Model Builder Program.

PROCESS LANE STRATEGIES



Process lane strategies showing data gathered for use by the Model Builder Program.

APPENDIX III

OBS CODE DATA FILE

OBS CODE DATA FILE

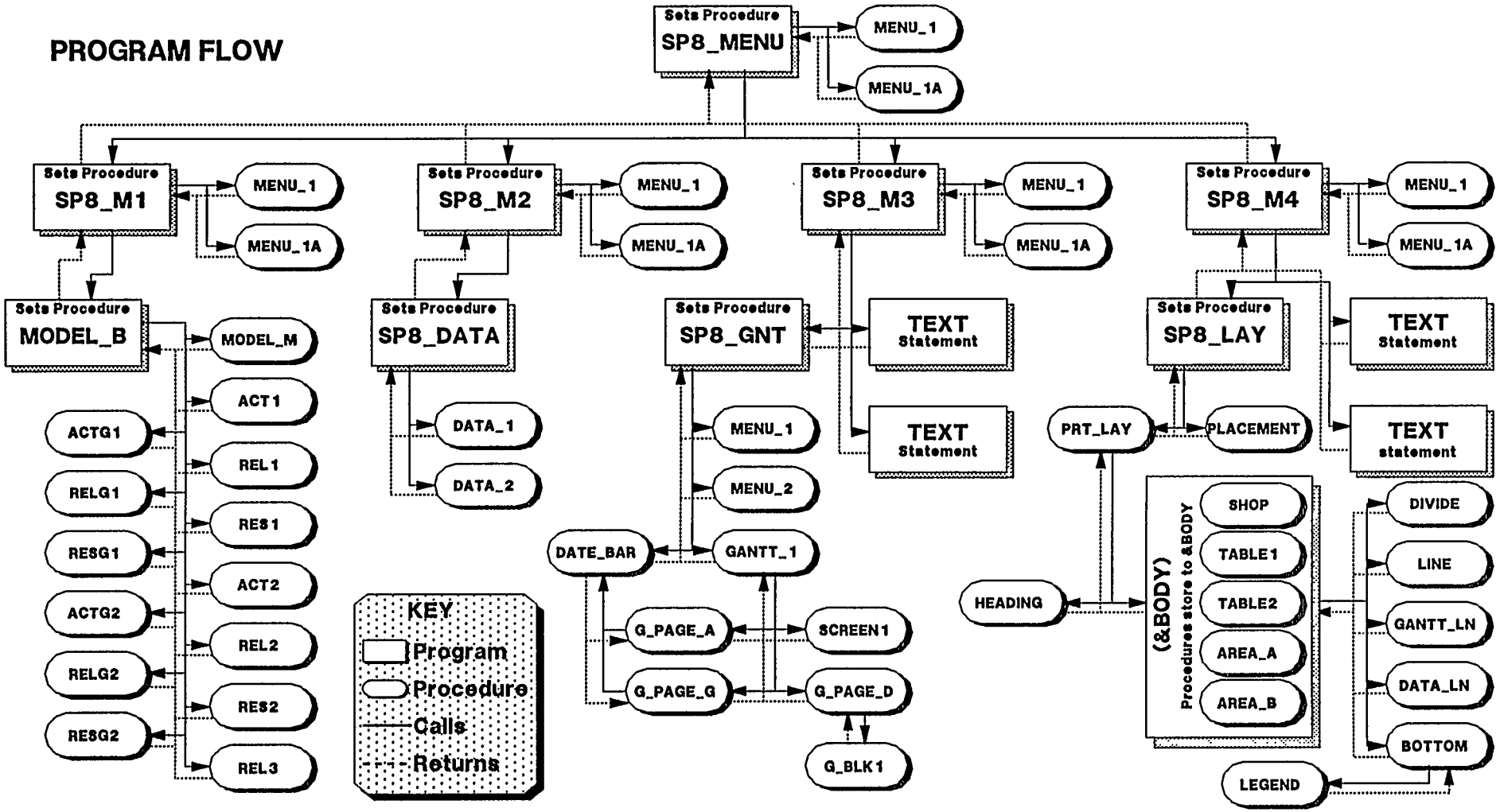
CODE FIELD 2 (C2)	DESCRIPTION (C2DESC)
1	Production
11	Steel Area's
12	Non-Steel Areas
111	Steel Fabrication
112	Steel Sub-Assembly
113	Steel Assembly
114	Steel Erection
1131	Steel Assembly - Flat Block
1132	Steel Assembly - Curved Block
1141	Erection - On-Board
1142	Erection - Grand Block Stacking
121	Pipe Shop
122	Vent Shop
123	Outfitting
124	Blast & Paint
1231	Outfitting - Pre-Blast
1232	Outfitting - Post Blast
12311	Outfitting - Pre-Blast (Inverted)
12312	Outfitting - Pre-Blast (Upright)

APPENDIX IV

PROGRAM LOGIC FLOW CHARTS
FOR

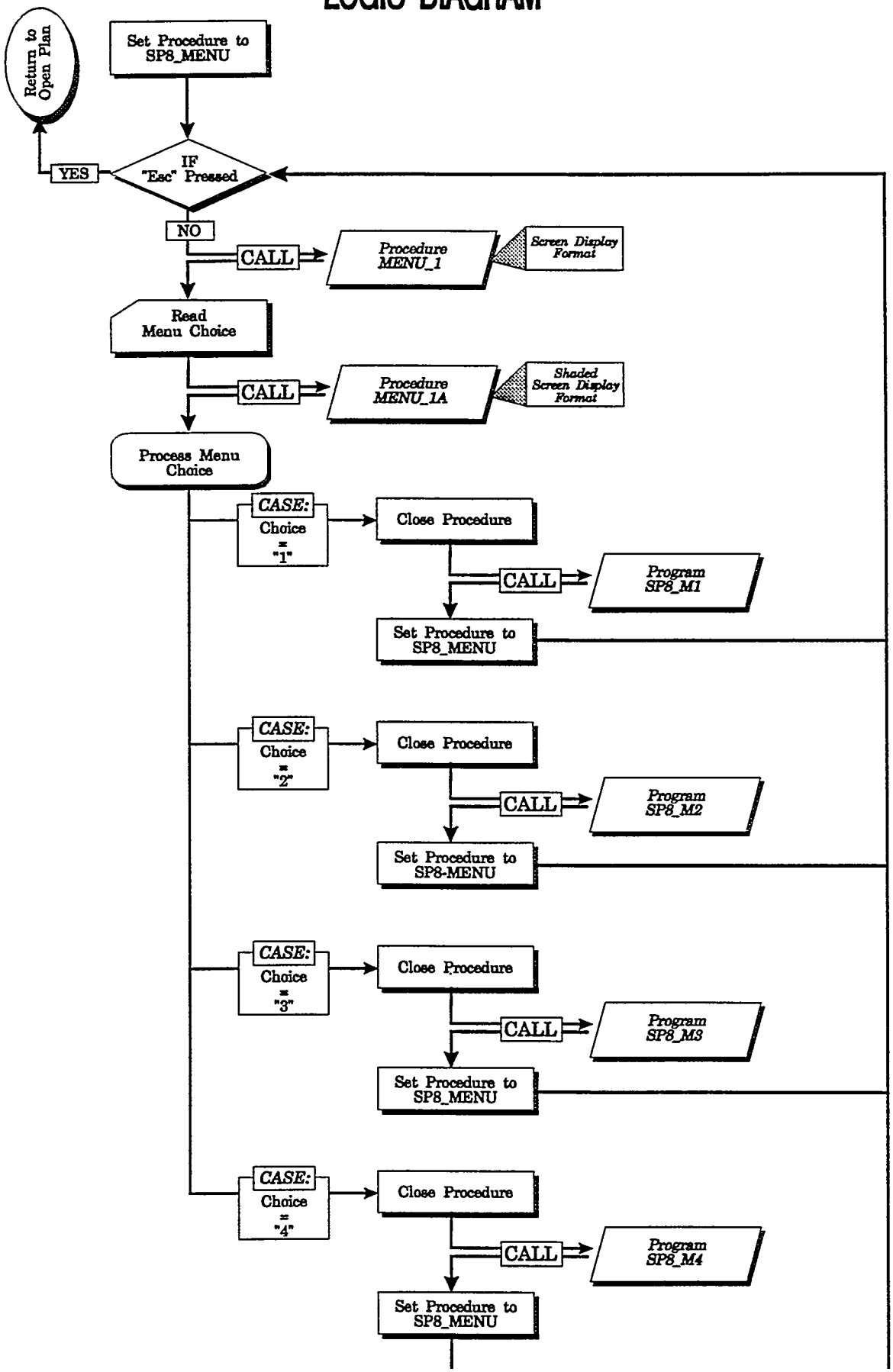
- **MENU SYSTEM**
- **MODEL BUILDER**
- **RESOURCE UTILIZATION CURVE GENERATION**
- **LAYDOWN CHART GENERATION**
- **GANIT CHART GENERATION**

PROGRAM FLOW



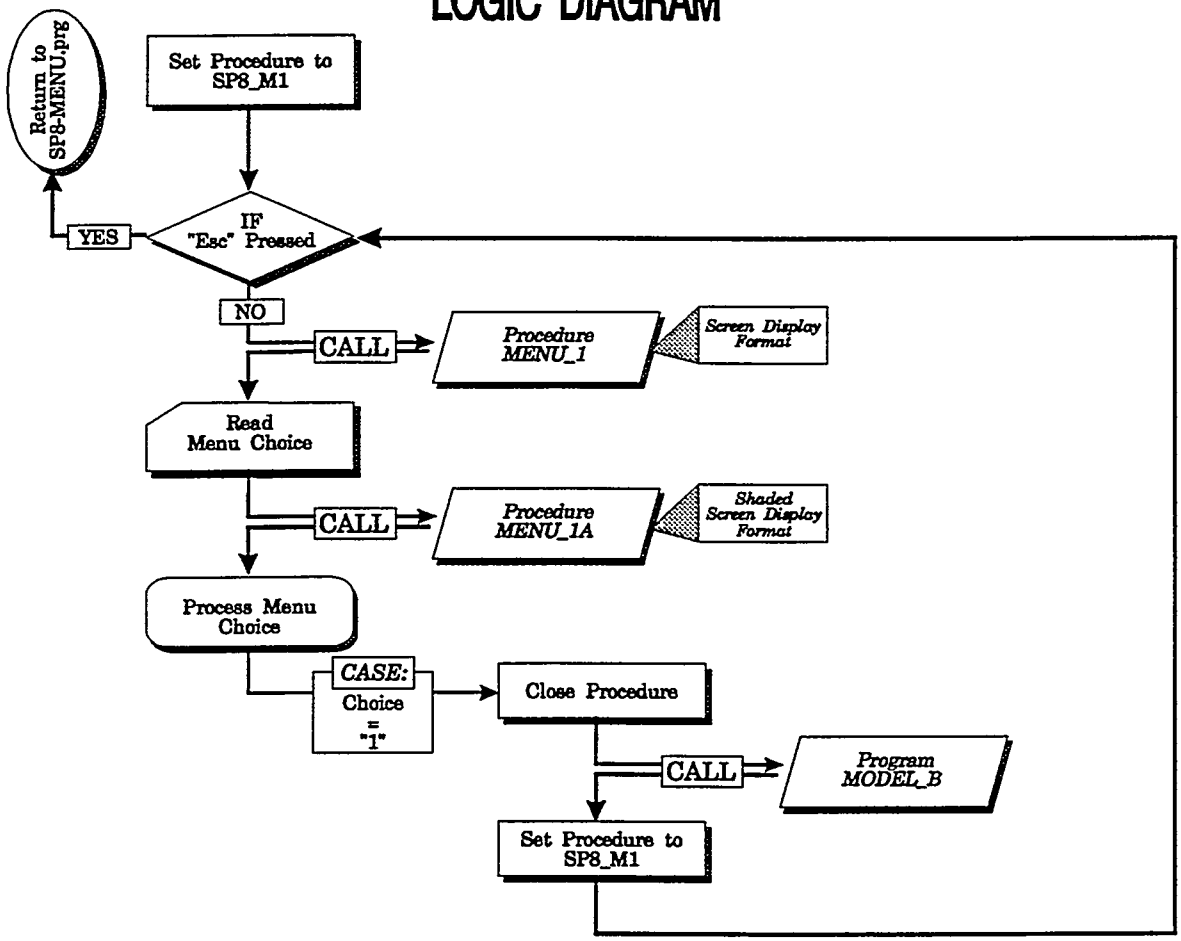
SP8_MENU.prg

LOGIC DIAGRAM

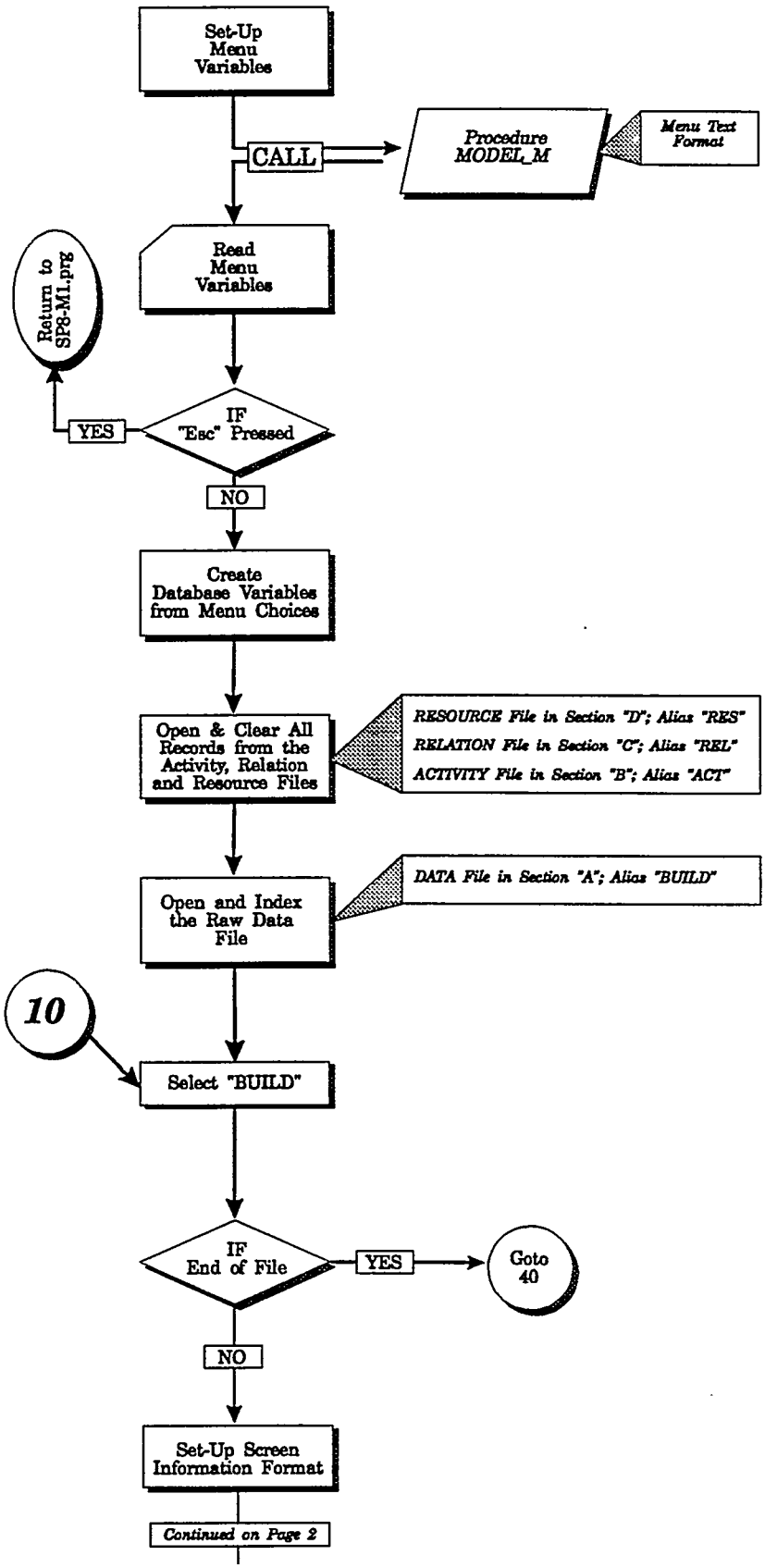


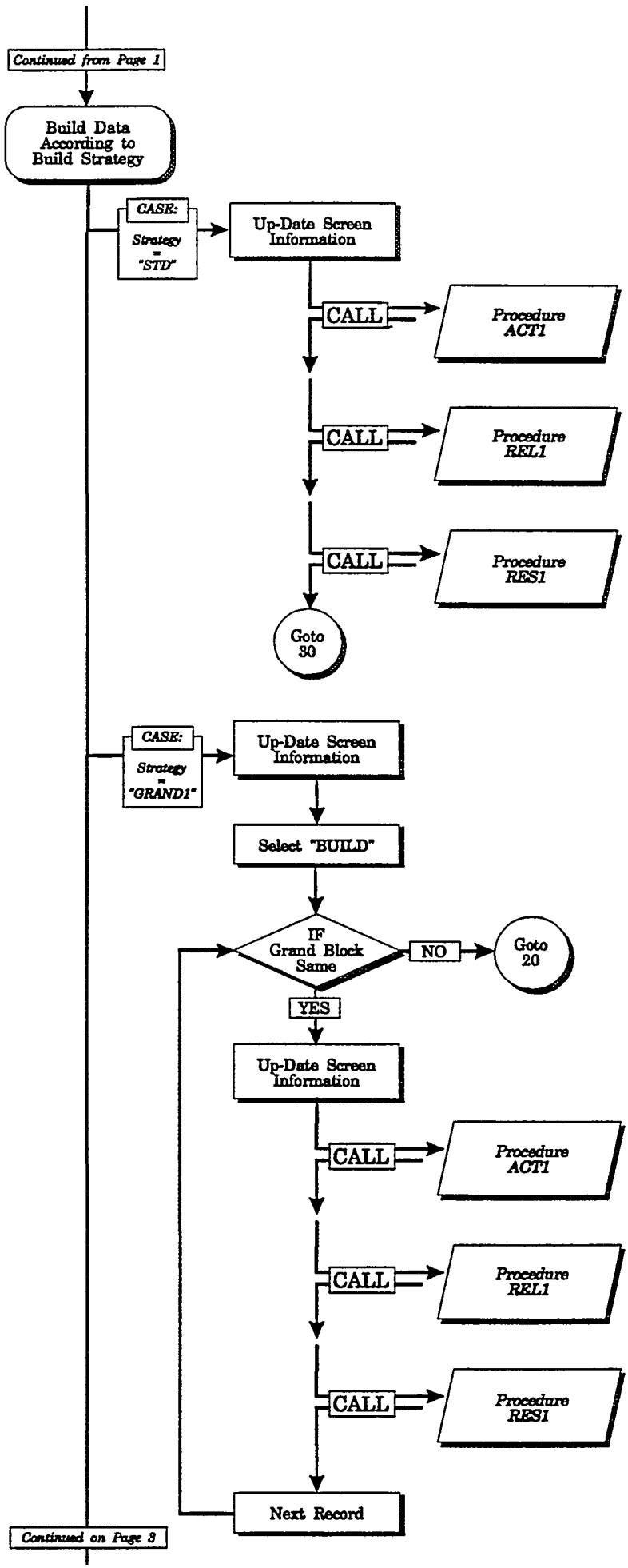
SP8_M1.prg

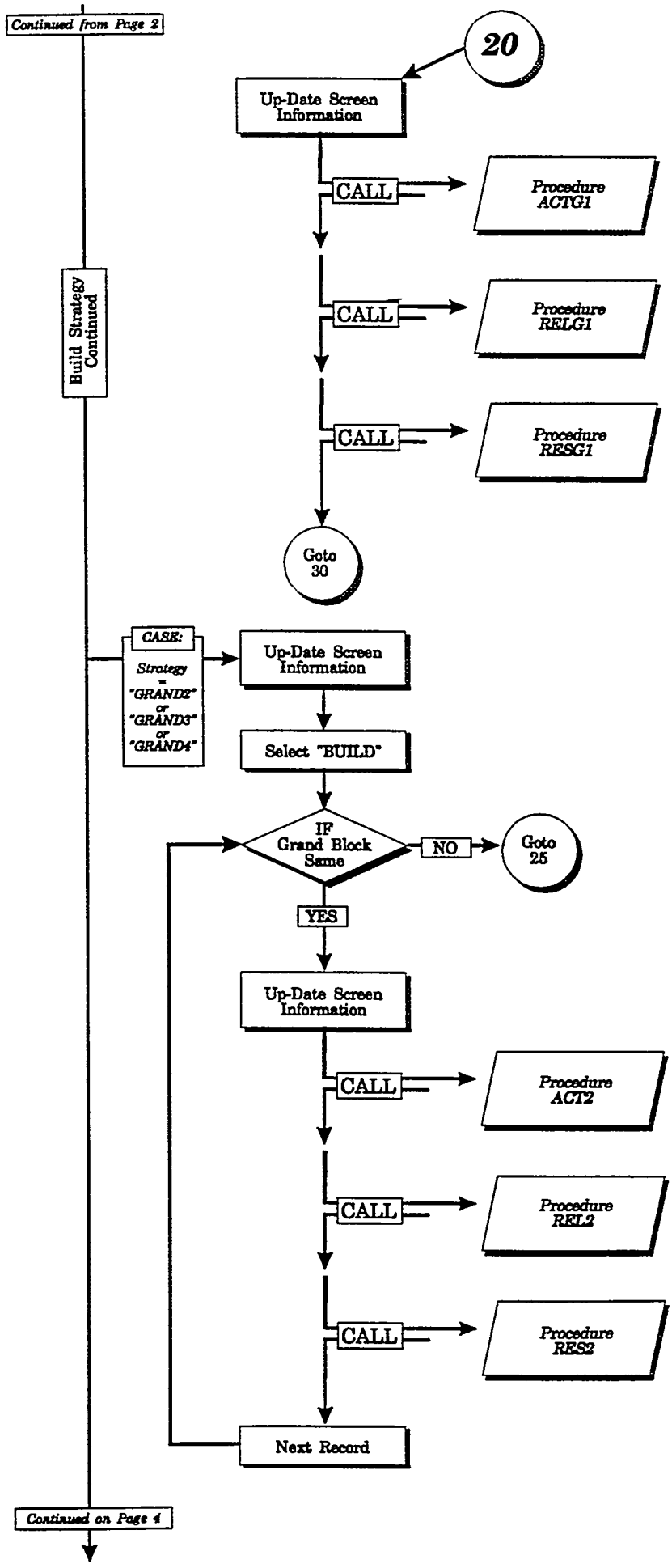
LOGIC DIAGRAM

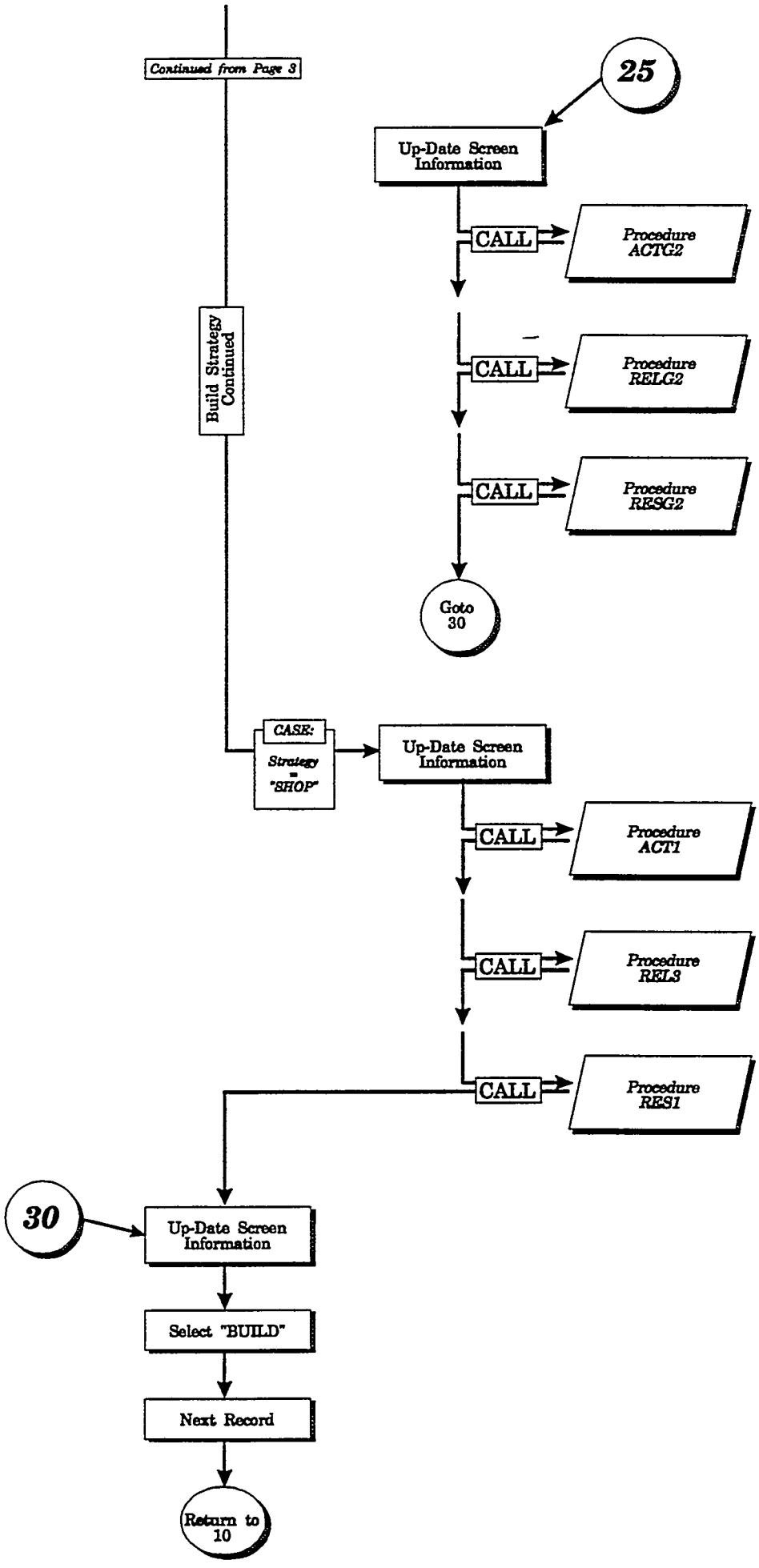


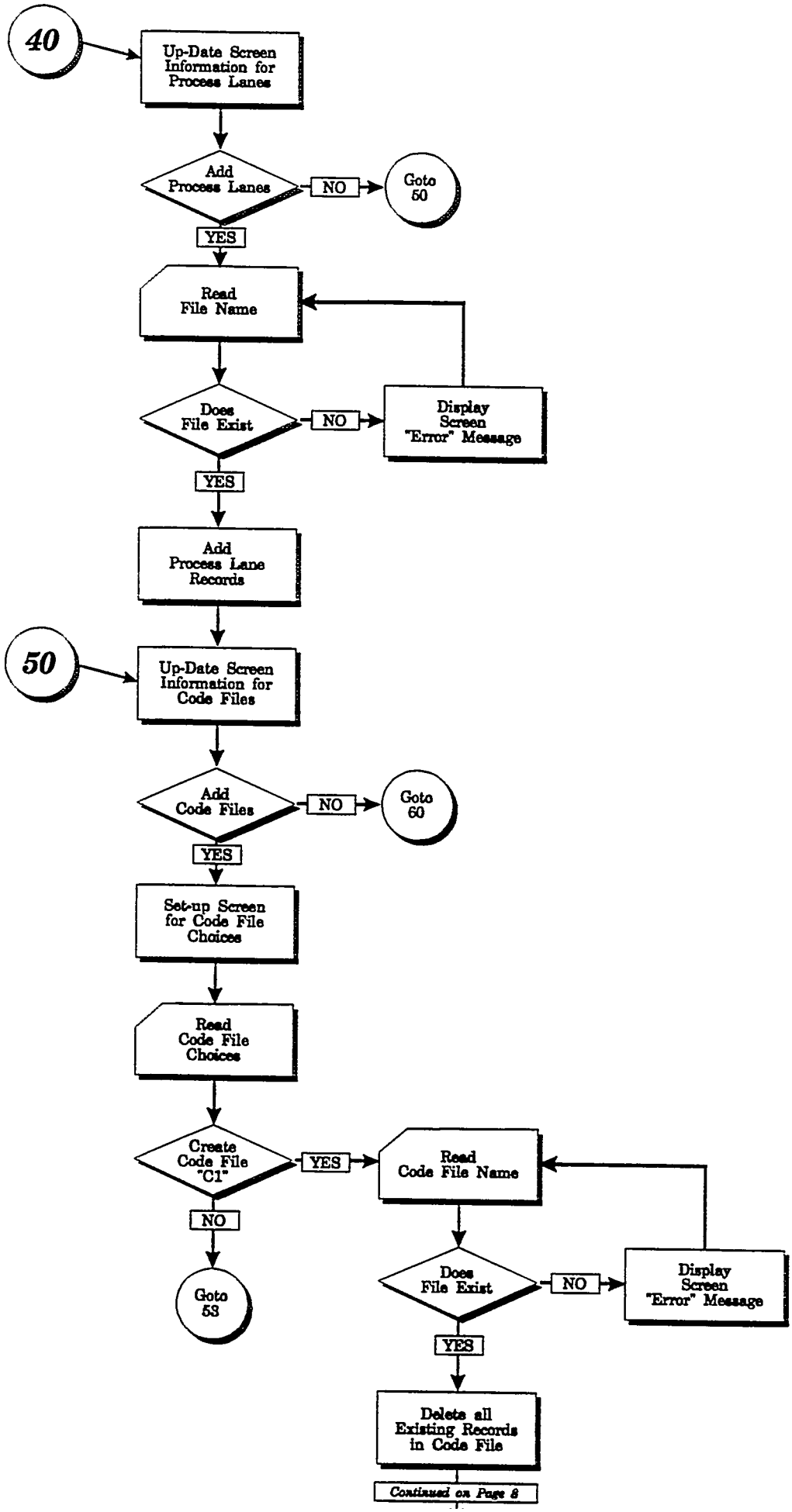
MODEL_B.prg
LOGIC DIAGRAM



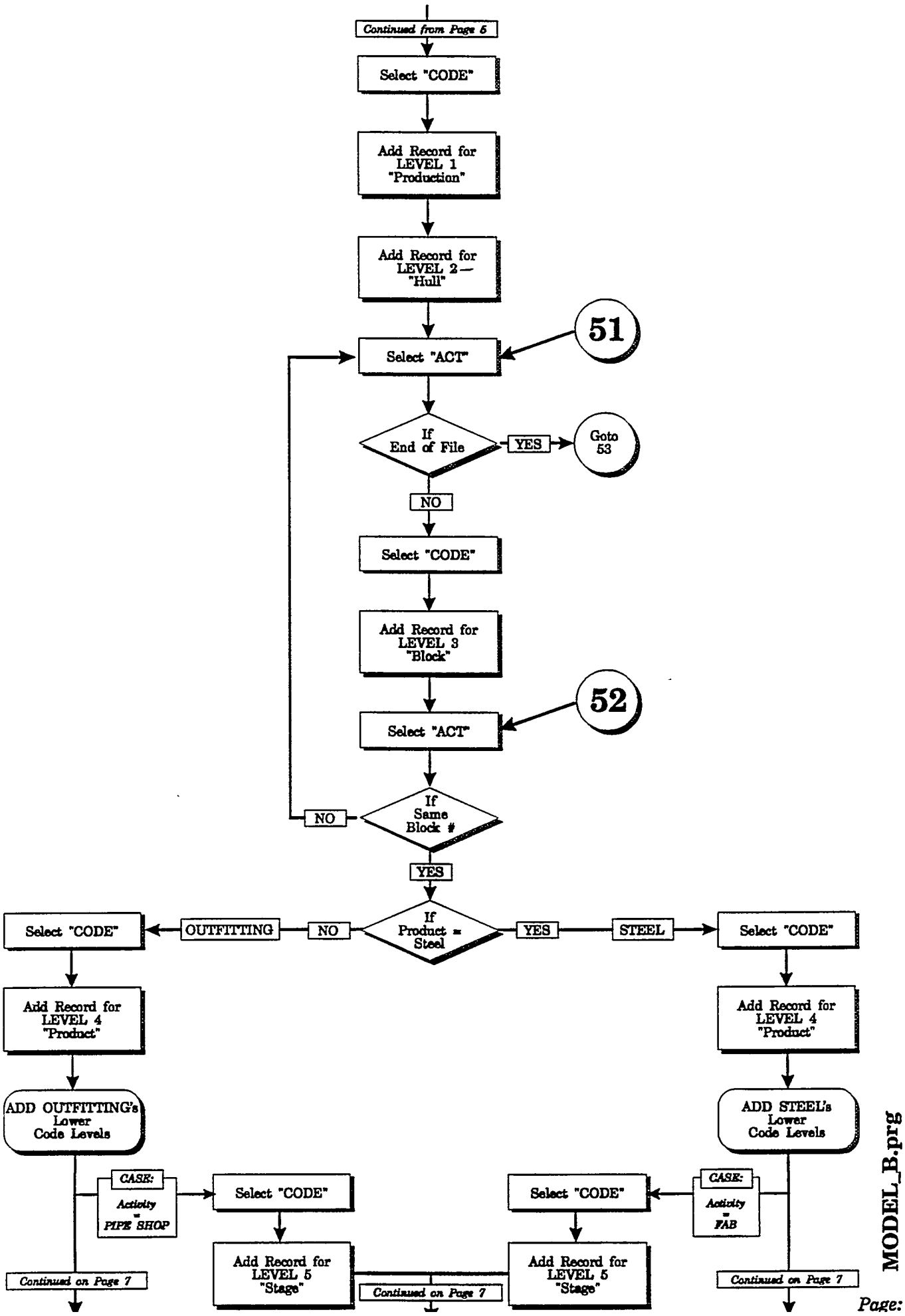

Continued from Page 1
Build Data According to Build Strategy
CASE: Strategy = "STD"
Up-Date Screen Information
CALL
Procedure ACT1
CALL
Procedure REL1
CALL
Procedure RES1
Goto 30
CASE: Strategy = "GRAND1"
Up-Date Screen Information
Select "BUILD"
IF Grand Block Same
NO
Goto 20
YES
Up-Date Screen Information
CALL
Procedure ACT1
CALL
Procedure REL1
CALL
Procedure RES1
Next Record
Continued on Page 3


Continued from Page 2
20
Up-Date Screen Information
CALL
Procedure ACTG1
CALL
Procedure RELG1
CALL
Procedure RESG1
Goto 30
Build Strategy Continued
CASE:
Strategy = "GRAND2" or "GRAND3" or "GRAND4"
Up-Date Screen Information
Select "BUILD"
IF Grand Block Same
NO
Goto 25
YES
Up-Date Screen Information
CALL
Procedure ACT2
CALL
Procedure REL2
CALL
Procedure RES2
Next Record
Continued on Page 4

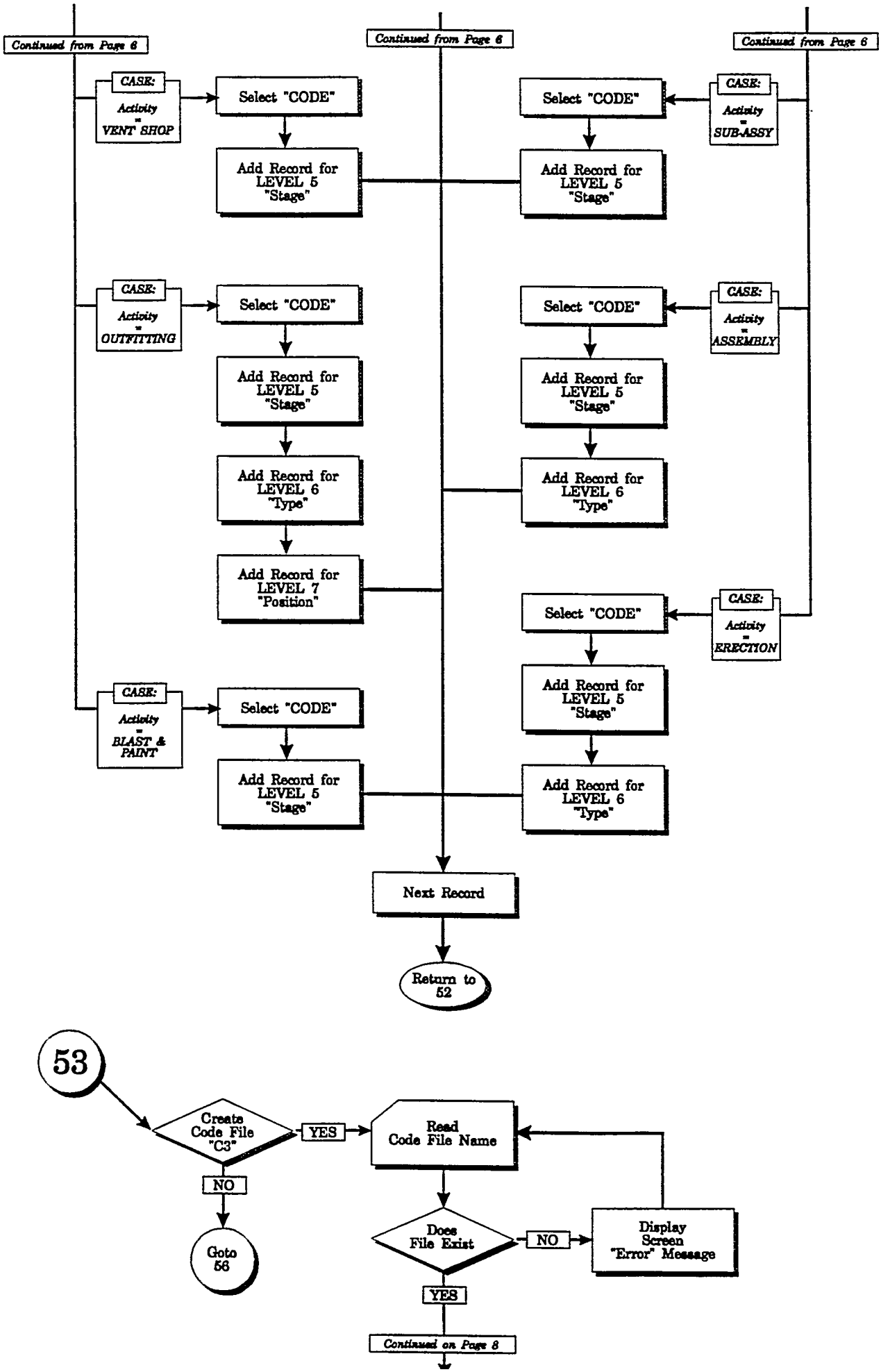

Continued from Page 3
25
Up-Date Screen Information
CALL
Procedure ACTG2
CALL
Procedure RELG2
CALL
Procedure RESG2
Goto 30
Build Strategy Continued
CASE: Strategy = "SHOP"
Up-Date Screen Information
CALL
Procedure ACT1
CALL
Procedure REL3
CALL
Procedure RES1
30
Up-Date Screen Information
Select "BUILD"
Next Record
Return to 10


40
Up-Date Screen Information for Process Lanes
Add Process Lanes
NO
Goto 50
YES
Read File Name
Does File Exist
NO
Display Screen "Error" Message
YES
Add Process Lane Records
50
Up-Date Screen Information for Code Files
Add Code Files
NO
Goto 60
YES
Set-up Screen for Code File Choices
Read Code File Choices
Create Code File "C1"
YES
Read Code File Name
NO
Goto 53
Does File Exist
NO
Display Screen "Error" Message
YES
Delete all Existing Records in Code File
Continued on Page 8

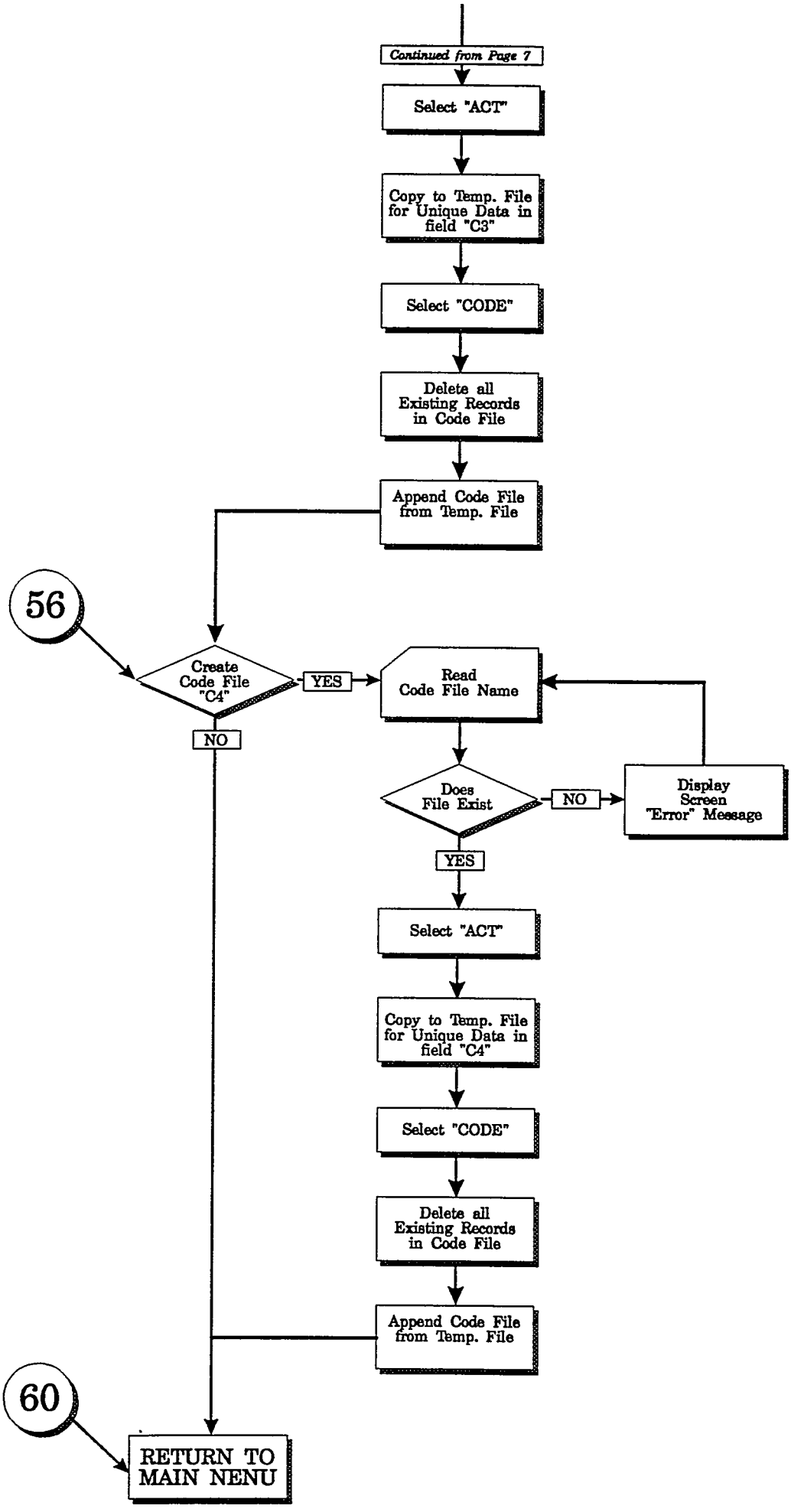
MODEL_B.prg


Continued from Page 6
Select "CODE"
Add Record for LEVEL 1 "Production"
Add Record for LEVEL 2 — "Hull"
51
Select "ACT"
If End of File
YES
Goto 53
NO
Select "CODE"
Add Record for LEVEL 3 "Block"
52
Select "ACT"
If Same Block #
NO
YES
If Product = Steel
OUTFITTING
NO
YES
STEEL
Select "CODE"
Add Record for LEVEL 4 "Product"
ADD OUTFITTING's Lower Code Levels
Select "CODE"
Add Record for LEVEL 4 "Product"
ADD STEEL's Lower Code Levels
CASE: Activity = PIPE SHOP
Select "CODE"
Add Record for LEVEL 5 "Stage"
CASE: Activity = FAB
Select "CODE"
Add Record for LEVEL 5 "Stage"
Continued on Page 7
Continued on Page 7
Continued on Page 7

MODEL_B.prg


Continued from Page 6
Continued from Page 6
Continued from Page 6
CASE: Activity = VENT SHOP
Select "CODE"
Add Record for LEVEL 5 "Stage"
Select "CODE"
CASE: Activity = SUB-ASSY
Add Record for LEVEL 5 "Stage"
CASE: Activity = OUTFITTING
Select "CODE"
Add Record for LEVEL 5 "Stage"
Add Record for LEVEL 6 "Type"
Add Record for LEVEL 7 "Position"
Select "CODE"
CASE: Activity = ASSEMBLY
Add Record for LEVEL 5 "Stage"
Add Record for LEVEL 6 "Type"
Select "CODE"
CASE: Activity = ERECTION
Add Record for LEVEL 5 "Stage"
Add Record for LEVEL 6 "Type"
CASE: Activity = BLAST & PAINT
Select "CODE"
Add Record for LEVEL 5 "Stage"
Next Record
Return to 52
53
Create Code File "C3"
YES
NO
Goto 56
Read Code File Name
Does File Exist
NO
Display Screen "Error" Message
YES
Continued on Page 8

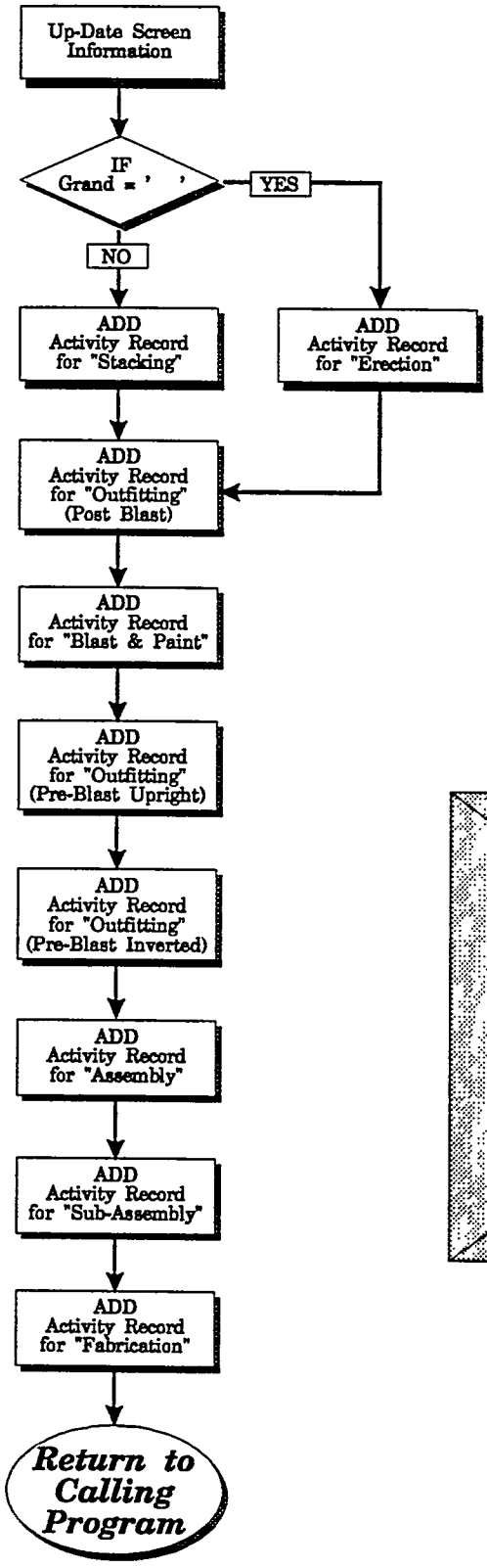
MODEL_B.prg


Continued from Page 7
Select "ACT"
Copy to Temp. File for Unique Data in field "C3"
Select "CODE"
Delete all Existing Records in Code File
Append Code File from Temp. File
56
Create Code File "C4"
YES
NO
Read Code File Name
Does File Exist
NO
Display Screen "Error" Message
YES
Select "ACT"
Copy to Temp. File for Unique Data in field "C4"
Select "CODE"
Delete all Existing Records in Code File
Append Code File from Temp. File
60
RETURN TO MAIN NENU

PROCEDURE ACT1

LOGIC DIAGRAM

Procedures ACTG1, ACT2 & ACTG2 are Similar



NOTE:

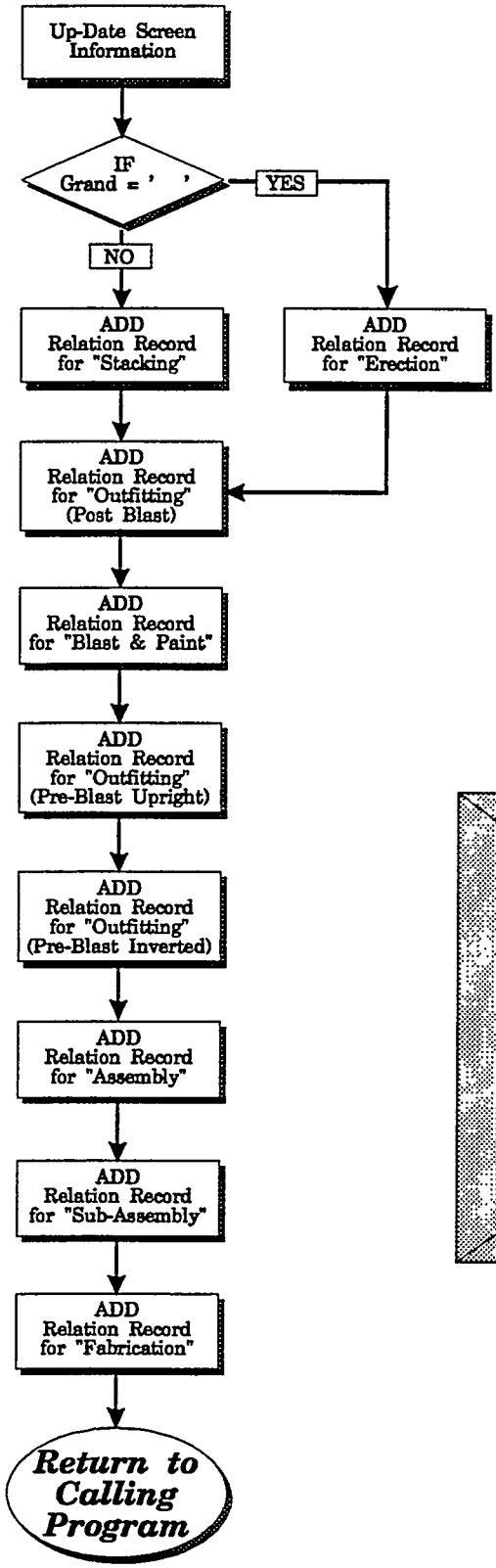
The following Fields are filled in for each Activity

ID - *ID Number*
DS - *Decription*
D - *Duration*
CAL - *Calendar*
C1 - *Code Field #1*
C2 - *Code Field #2*
C3 - *Code Field #3*
C4 - *Code Field #4*

PROCEDURE REL1

LOGIC DIAGRAM

Procedures RELG1, REL2, RELG2 & REL3 are Similar



NOTE:

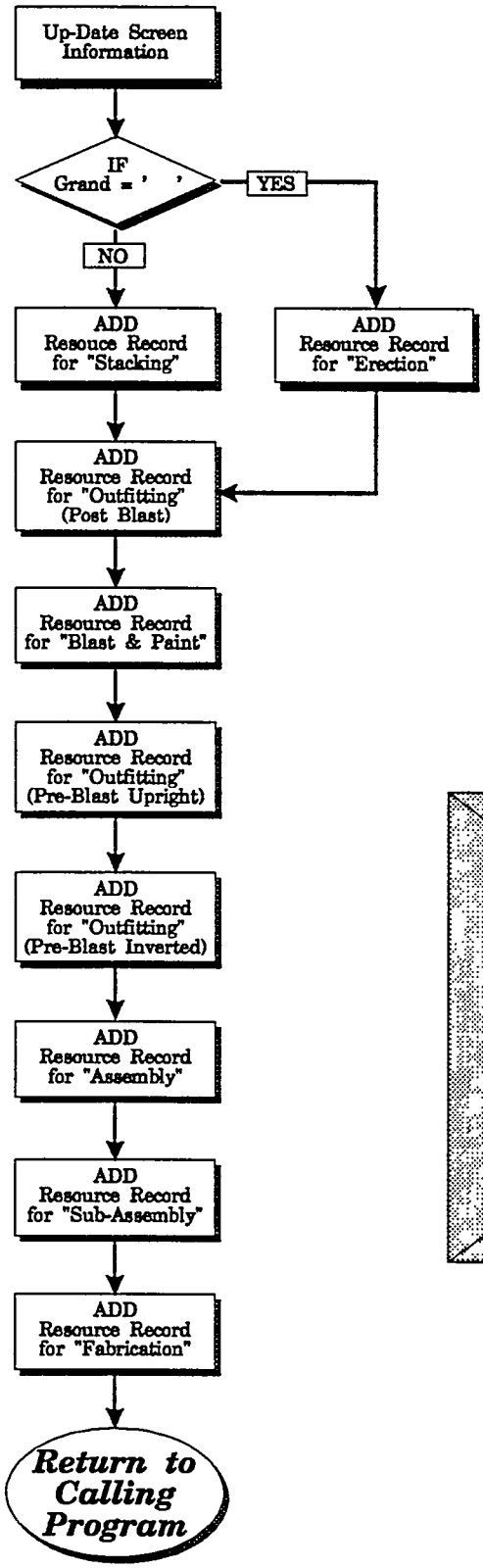
The following Fields are filled in for each Relation

ID - *ID Number*
PRED - *Predecessor ID Number*
TYPE - *Relation Type; "FS", "FF", "SF", "SS"*
LAG - *Days between Relation*

PROCEDURE RES1

LOGIC DIAGRAM

Procedures RESG1, RES2 & RESG2 are Similar



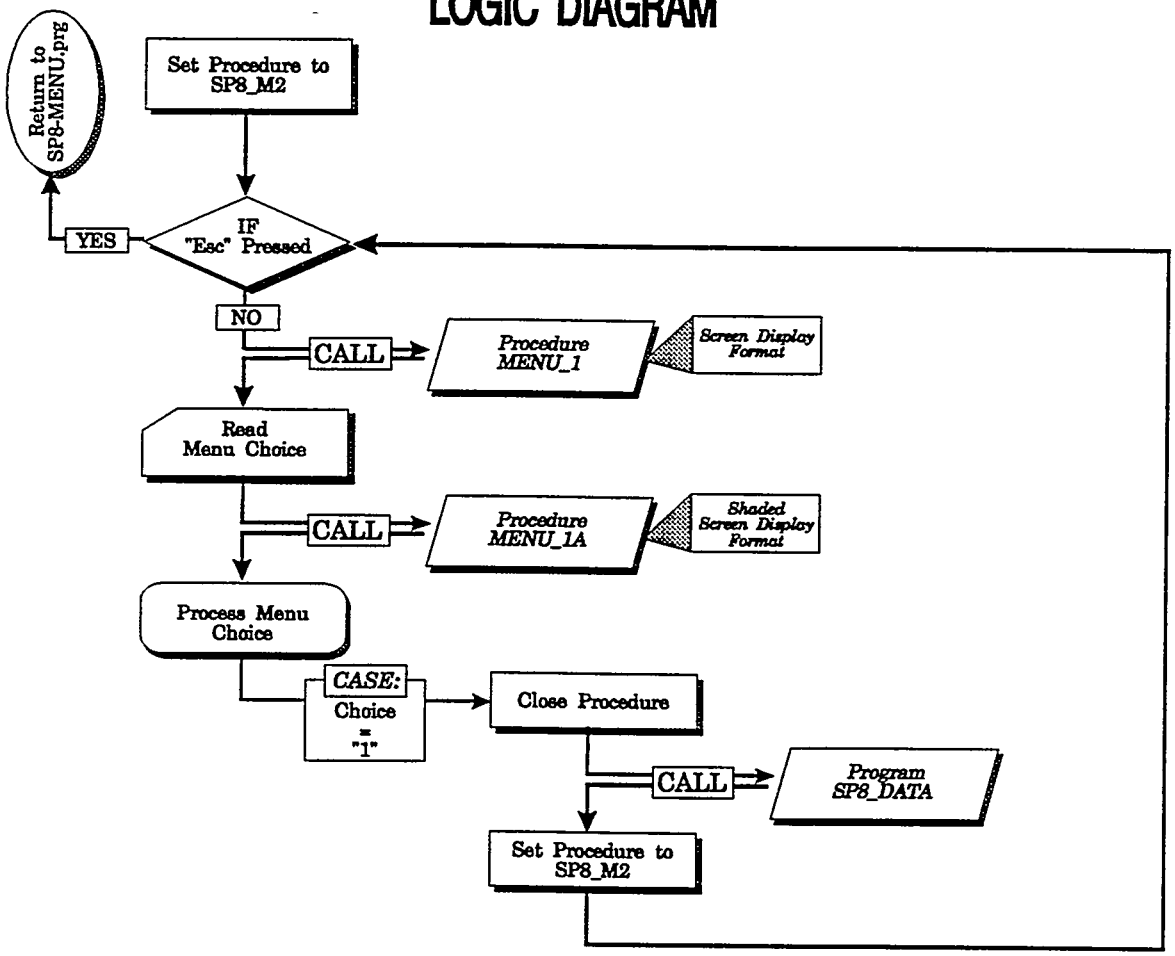
NOTE:

The following Fields are filled in for each Resource

ID - *ID Number*
RESCODE - *Code Name for the Resource*
LEVEL - *Quantity of Resouce*
LEVTYPE - *How to spread Resource*

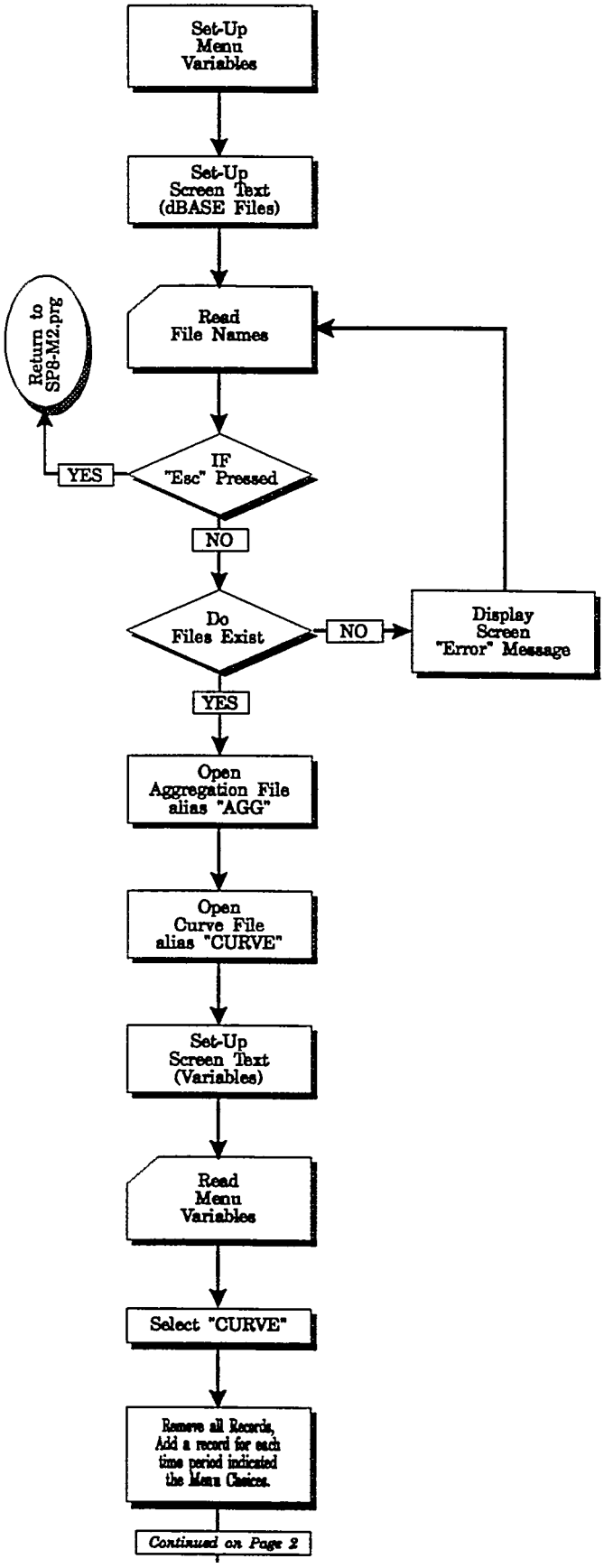
SP8_M2.prg

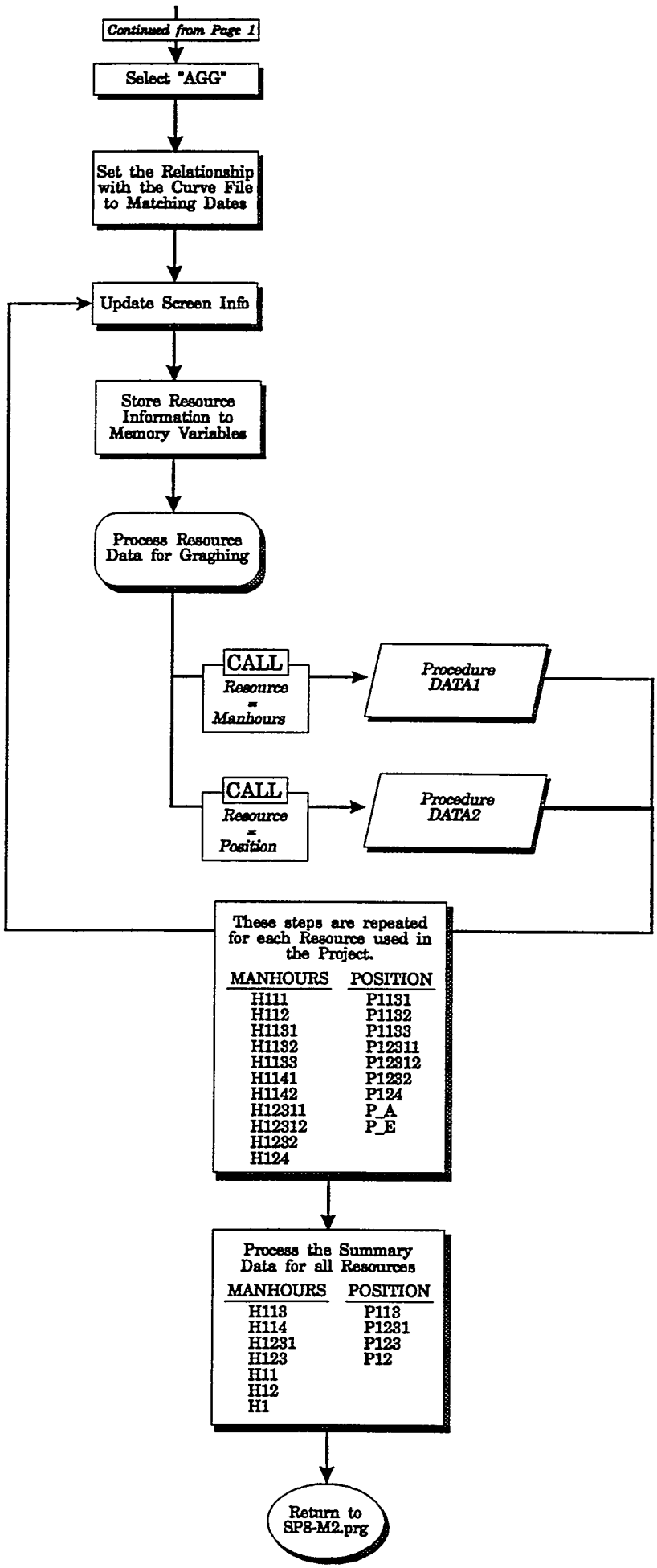
LOGIC DIAGRAM



SP8_DATA.prg

LOGIC DIAGRAM

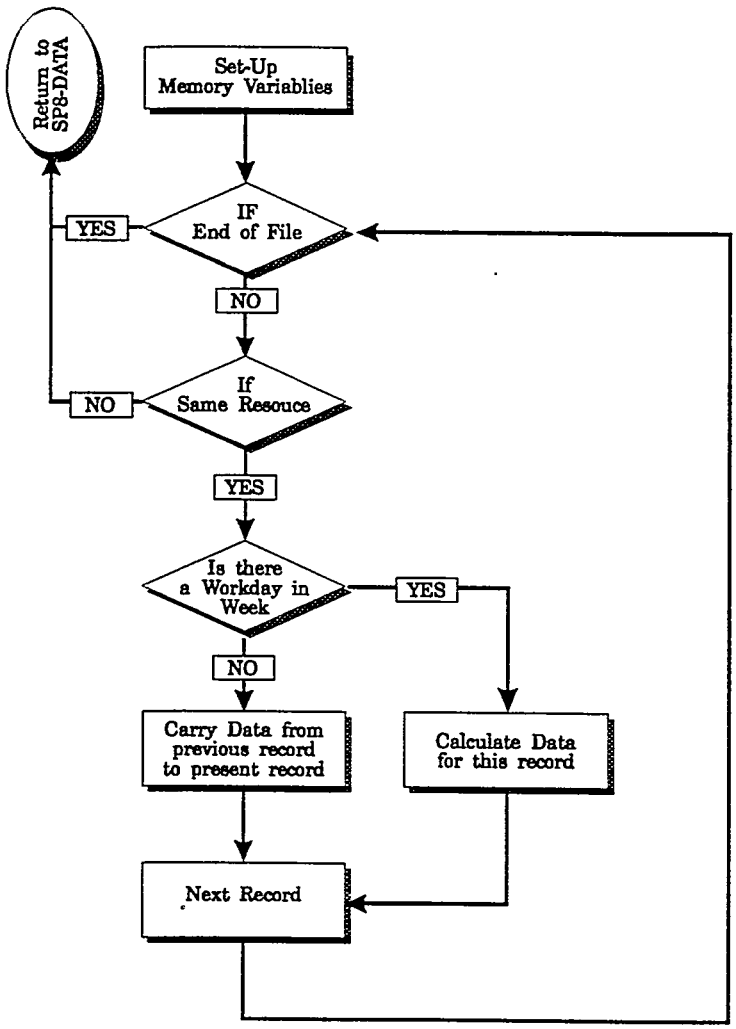



Continued from Page 1
Select "AGG"
Set the Relationship with the Curve File to Matching Dates
Update Screen Info
Store Resource Information to Memory Variables
Process Resource Data for Graghing
CALL
Resource = Manhours
Procedure DATA1
CALL
Resource = Position
Procedure DATA2
These steps are repeated for each Resource used in the Project.
MANHOURS
H111
H112
H1131
H1132
H1133
H1141
H1142
H12311
H12312
H1232
H124
POSITION
P1131
P1132
P1133
P12311
P12312
P1232
P124
P_A
P_E
Process the Summary Data for all Resources
MANHOURS
H113
H114
H1231
H123
H11
H12
H1
POSITION
P113
P1231
P123
P12
Return to SP8-M2.prg

PROCEDURE DATA1

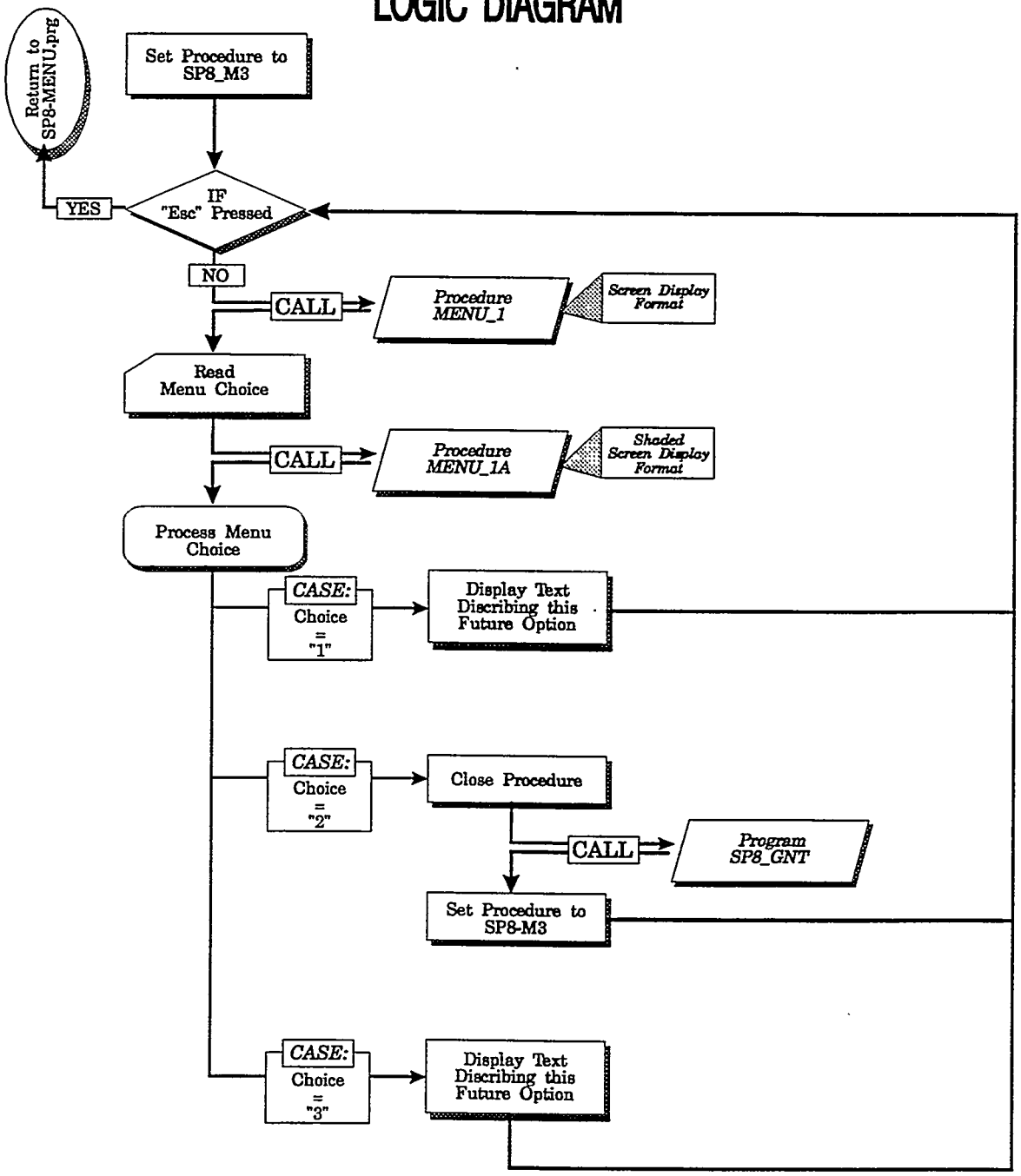
LOGIC DIAGRAM

Procedure DATA2 is similar



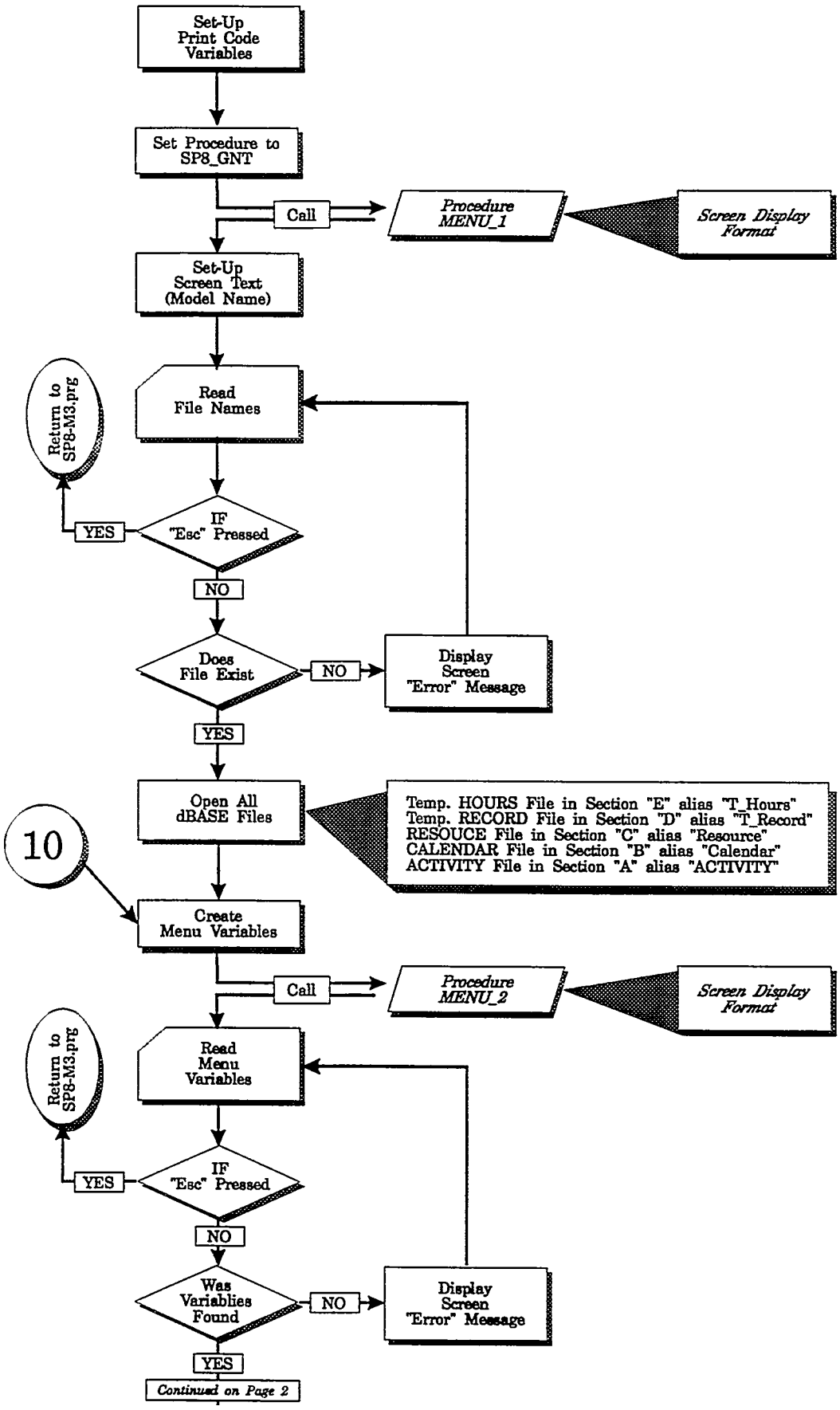
SP8_M3.prg

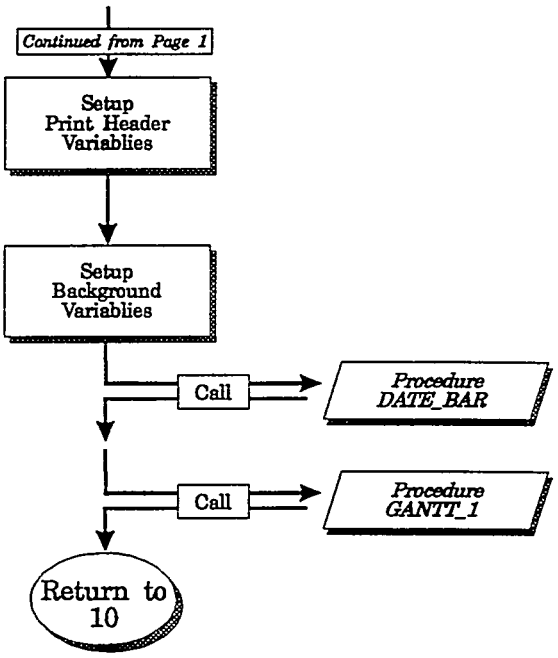
LOGIC DIAGRAM



SP8_GNT.prg

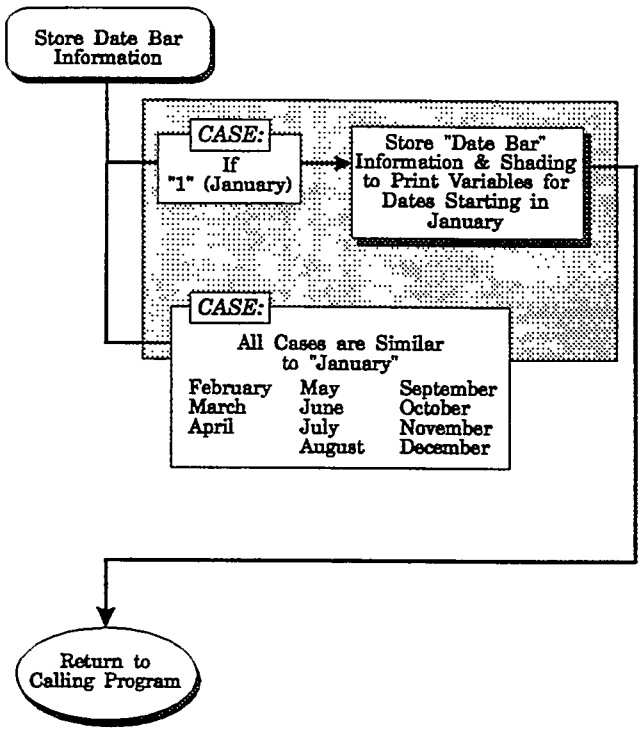
LOGIC DIAGRAM




Continued from Page 1
Setup
Print Header
Variablies
Setup
Background
Variablies
Call
Procedure
DATE_BAR
Call
Procedure
GANTT_1
Return to
10

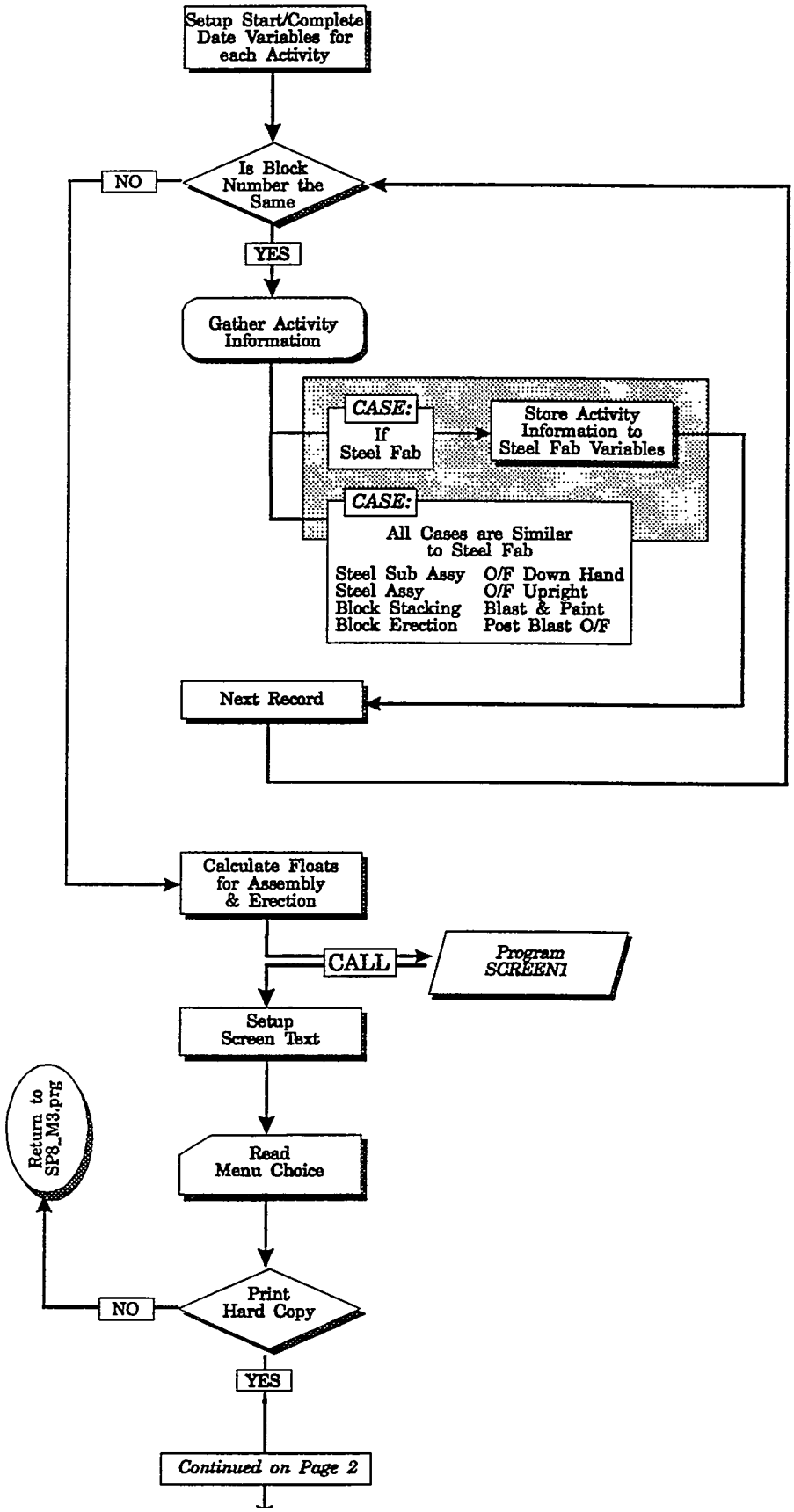
PROCEDURE DATE_BAR

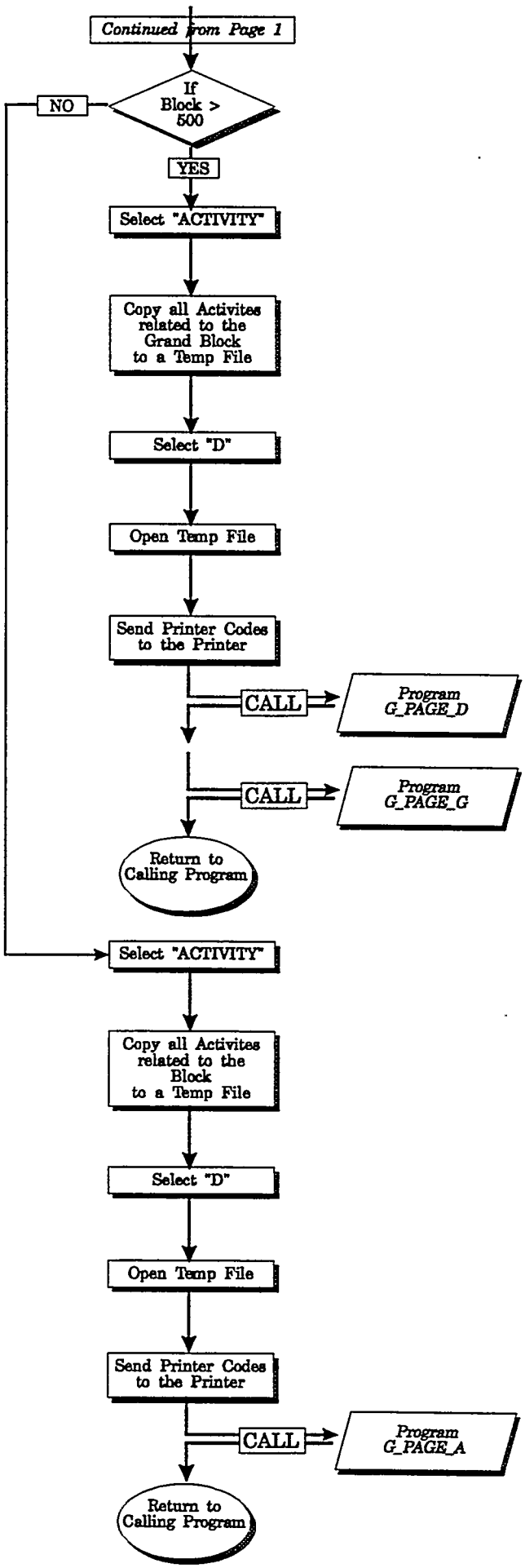
LOGIC DIAGRAM



PROCEDURE GANTT_1

LOGIC DIAGRAM



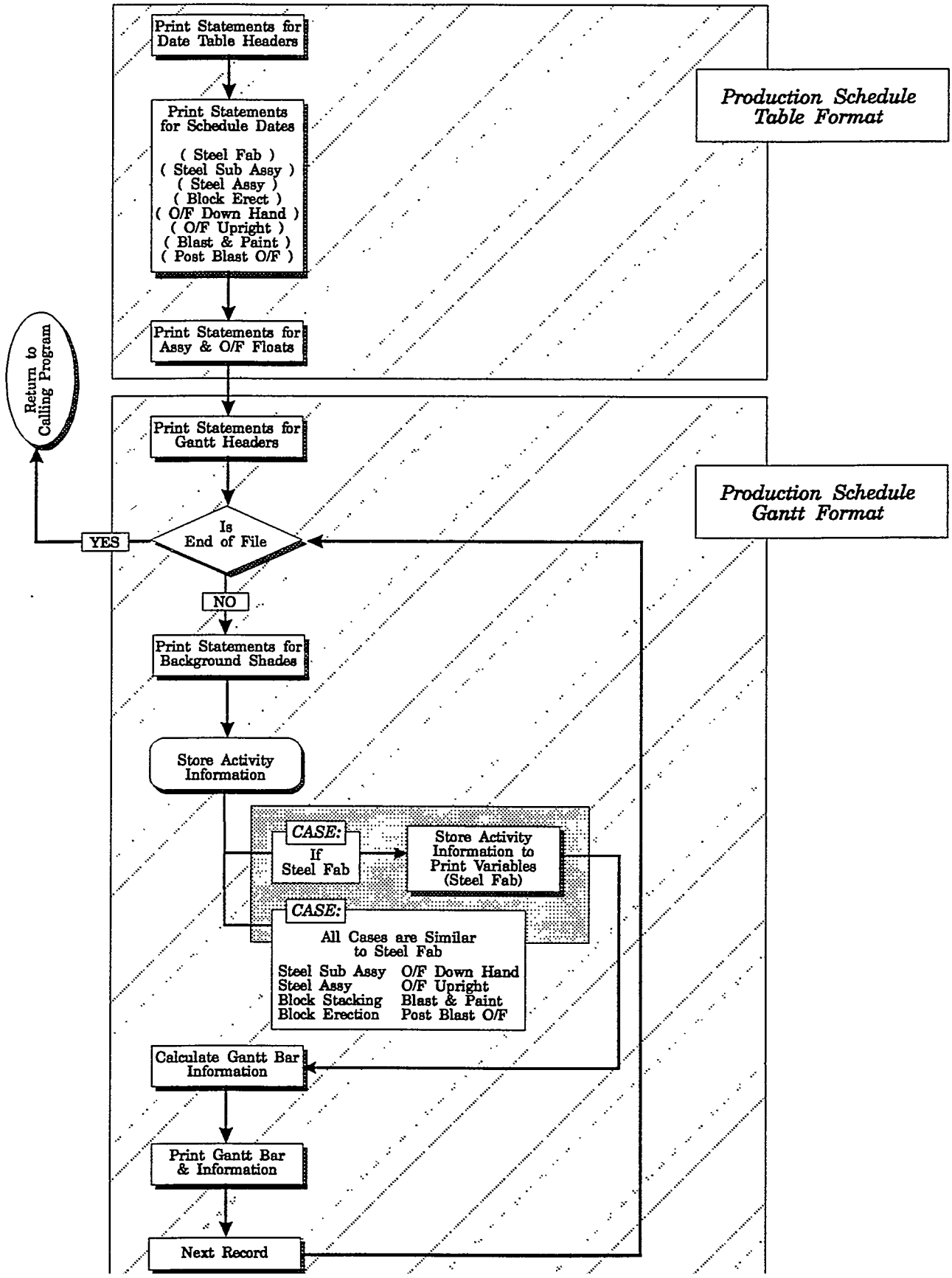


Procedure GANTT_1

PROCEDURE G_PAGE_A

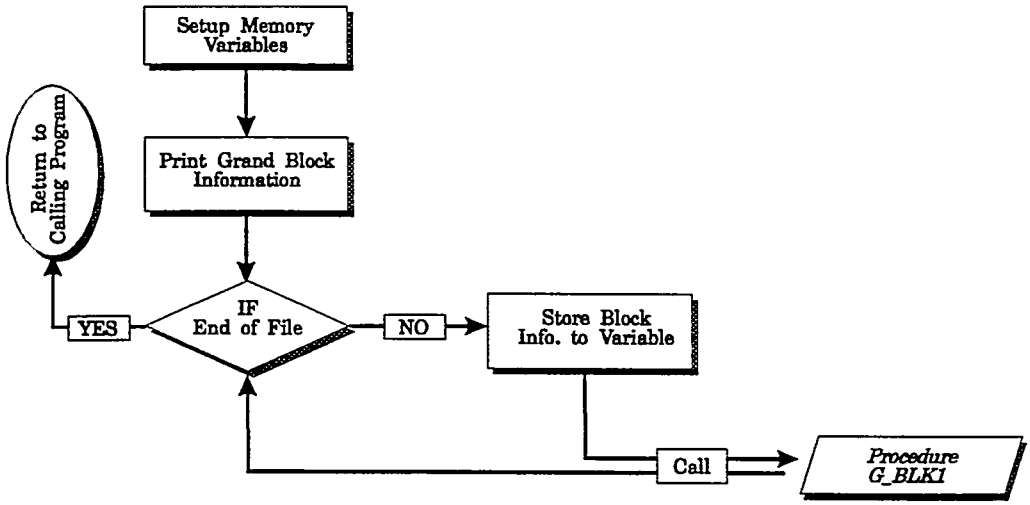
LOGIC DIAGRAM

Procedure G_PAGE_G, G_BLK1 & SCREEN1 are Similar



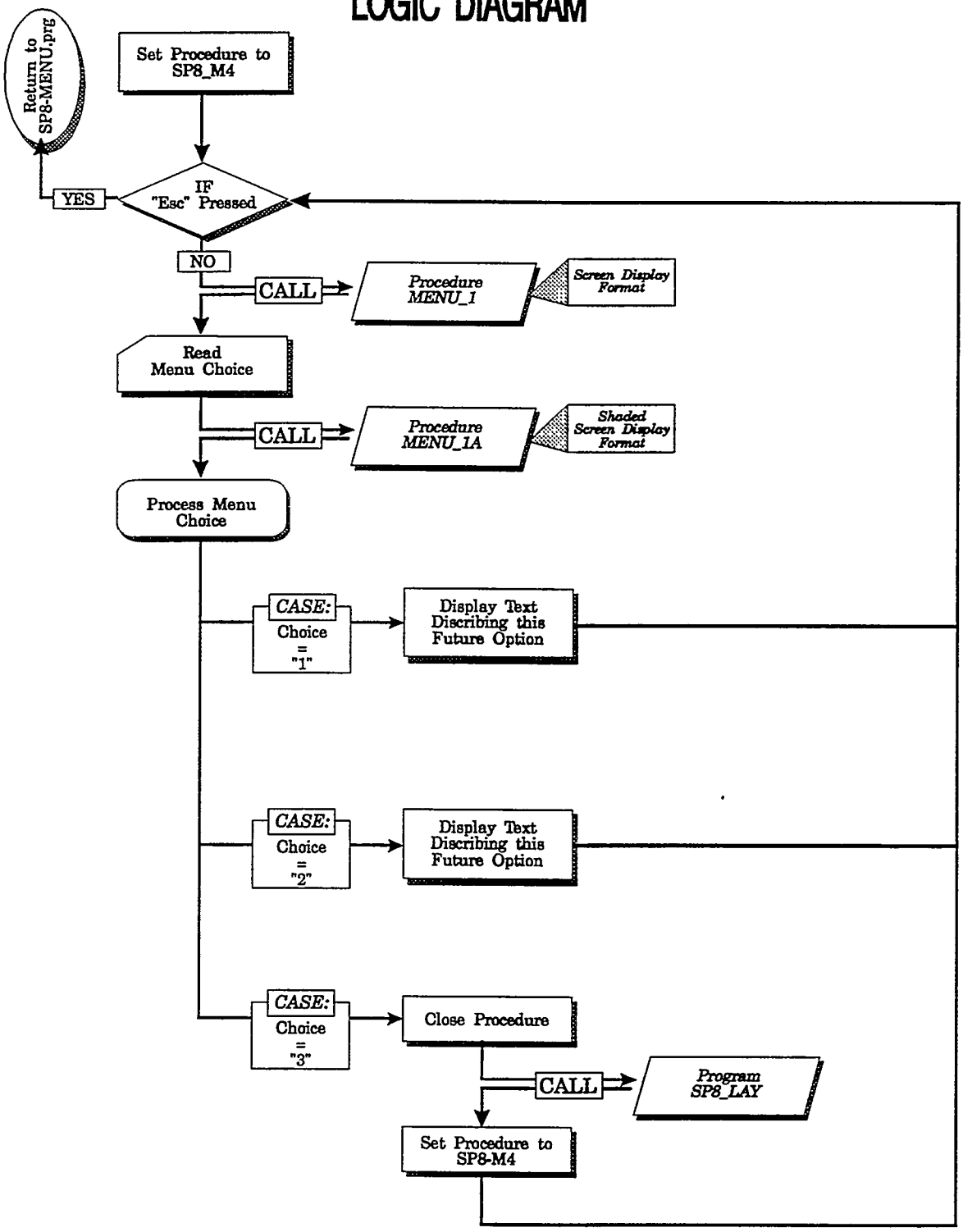
PROCEDURE G_PAGE_D

LOGIC DIAGRAM



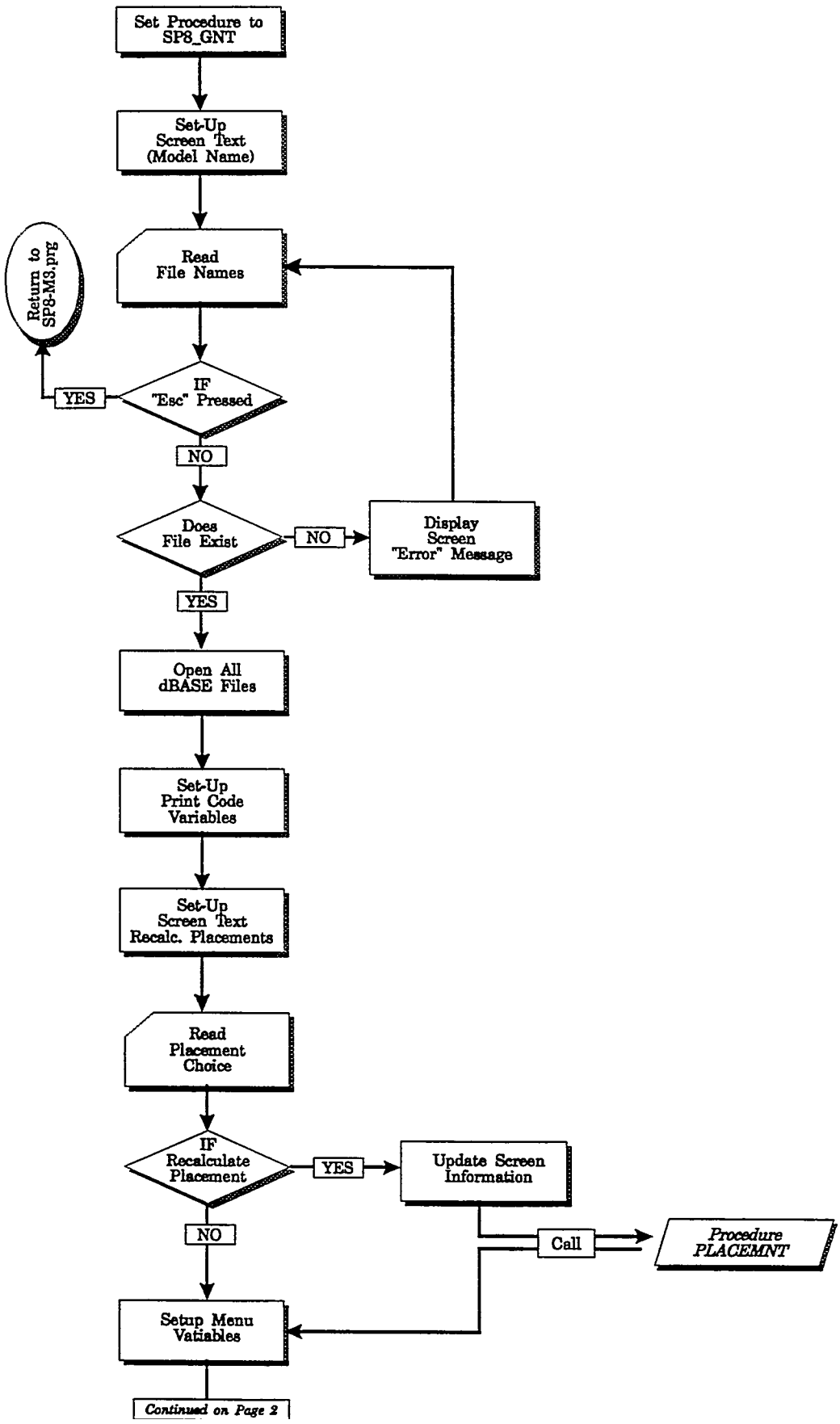
SP8_M4.prg

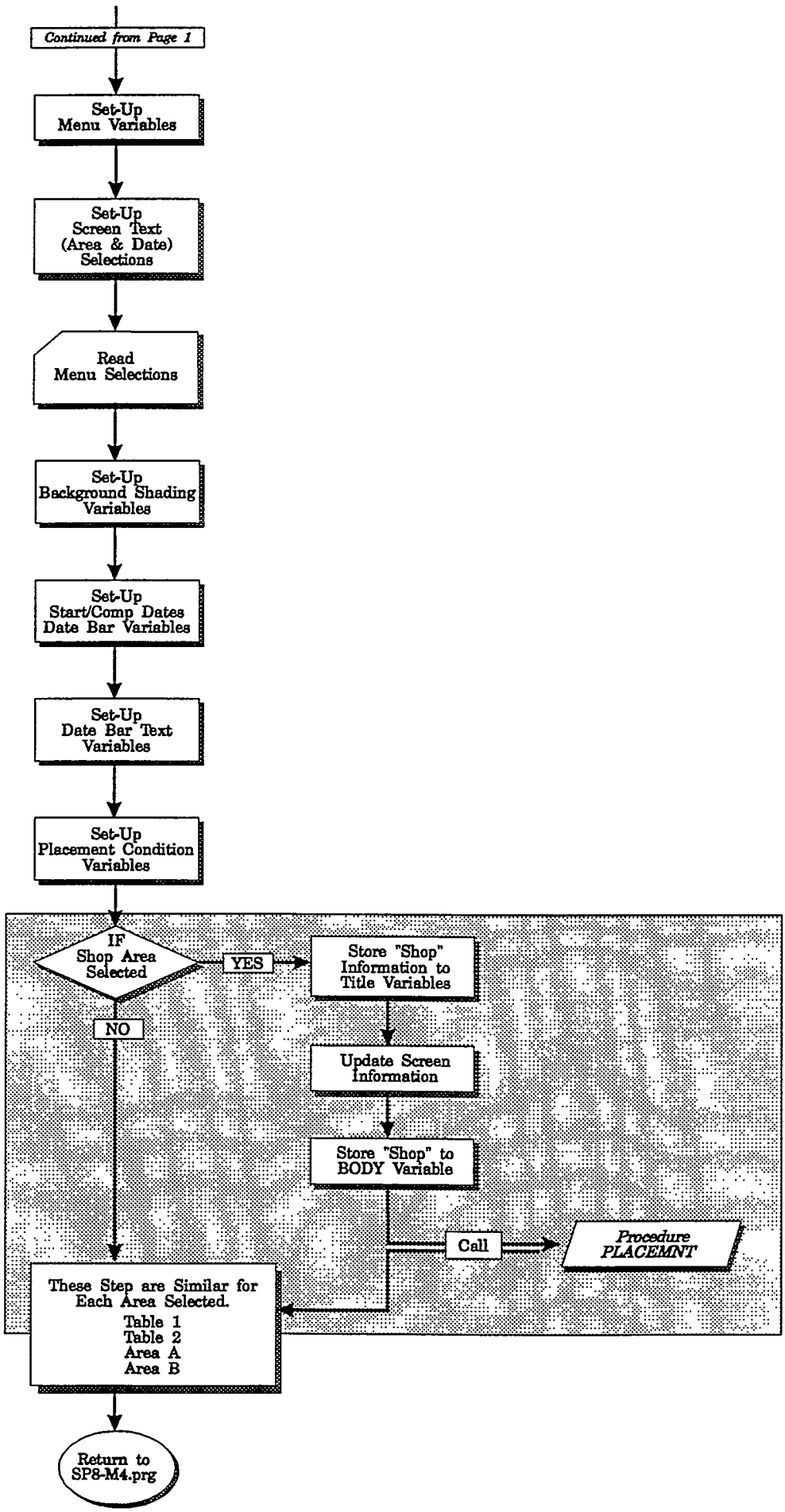
LOGIC DIAGRAM



SP8_LAY.prg

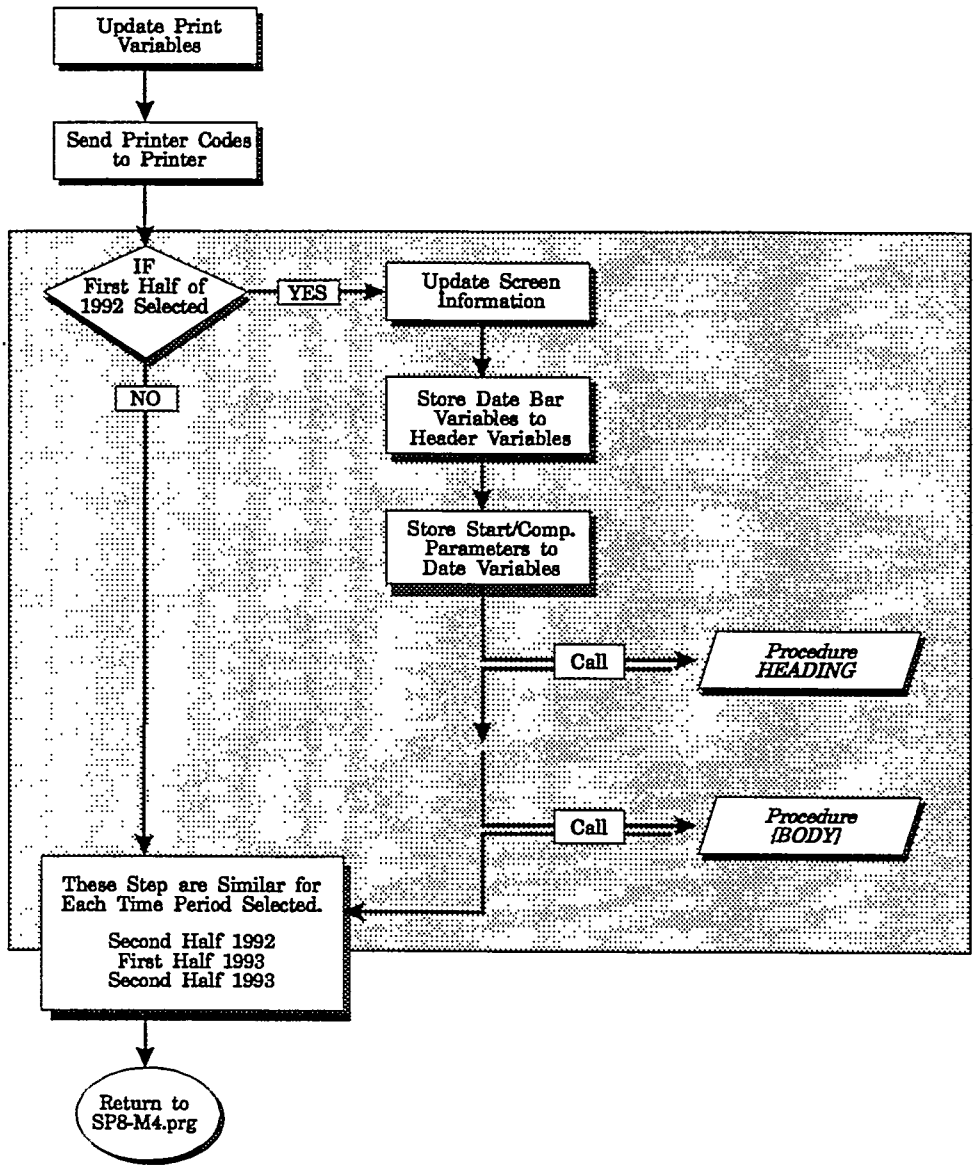
LOGIC DIAGRAM




Continued from Page 1
Set-Up Menu Variables
Set-Up Screen Text (Area & Date) Selections
Read Menu Selections
Set-Up Background Shading Variables
Set-Up Start/Comp Dates Date Bar Variables
Set-Up Date Bar Text Variables
Set-Up Placement Condition Variables
IF Shop Area Selected
YES
Store "Shop" Information to Title Variables
Update Screen Information
Store "Shop" to BODY Variable
Call
Procedure PLACEMNT
NO
These Step are Similar for Each Area Selected.
Table 1
Table 2
Area A
Area B
Return to SP8-M4.prg

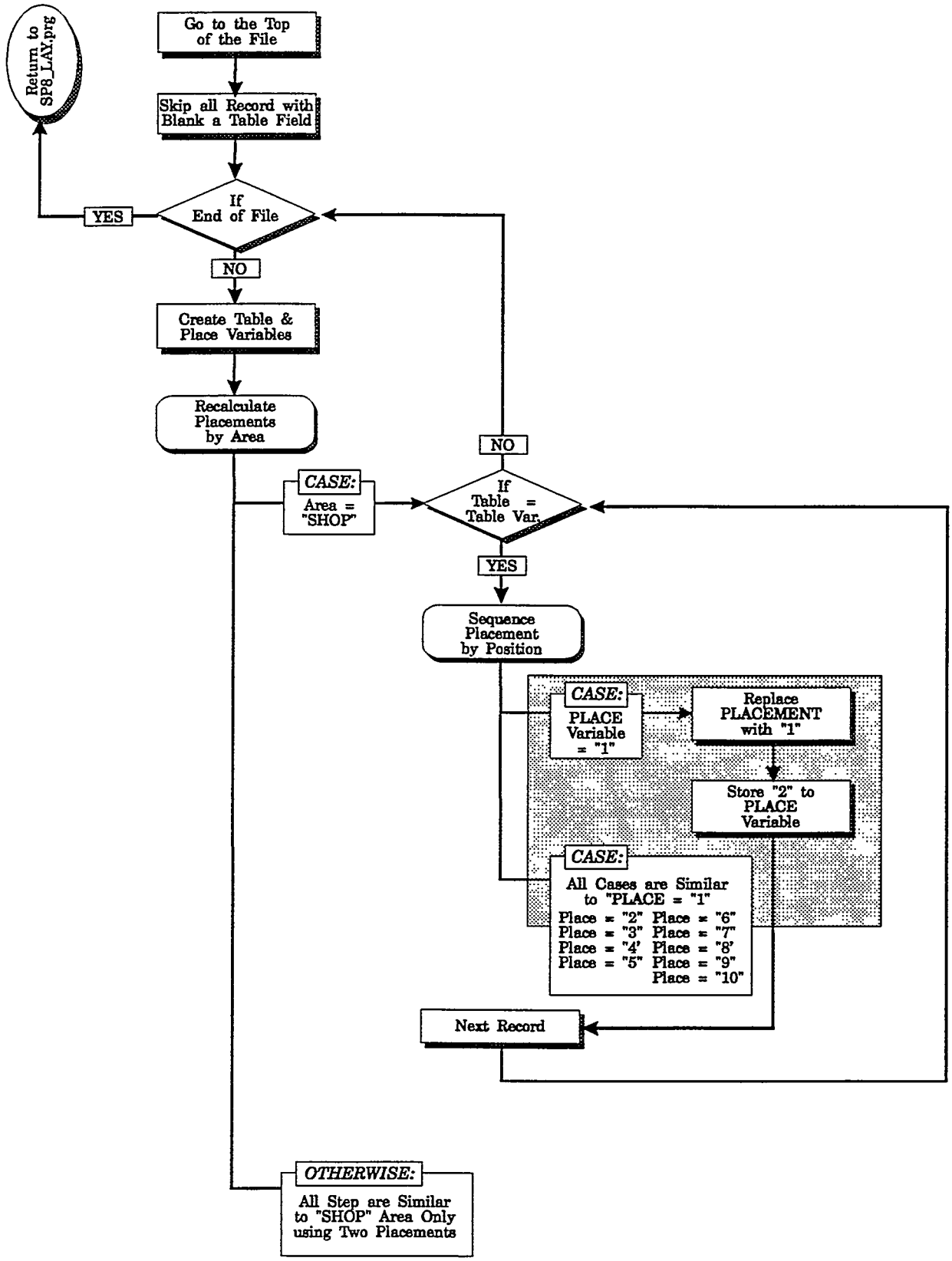
PROCEDURE PRT_LAY

LOGIC DIAGRAM



PROCEDURE PLACEMNT

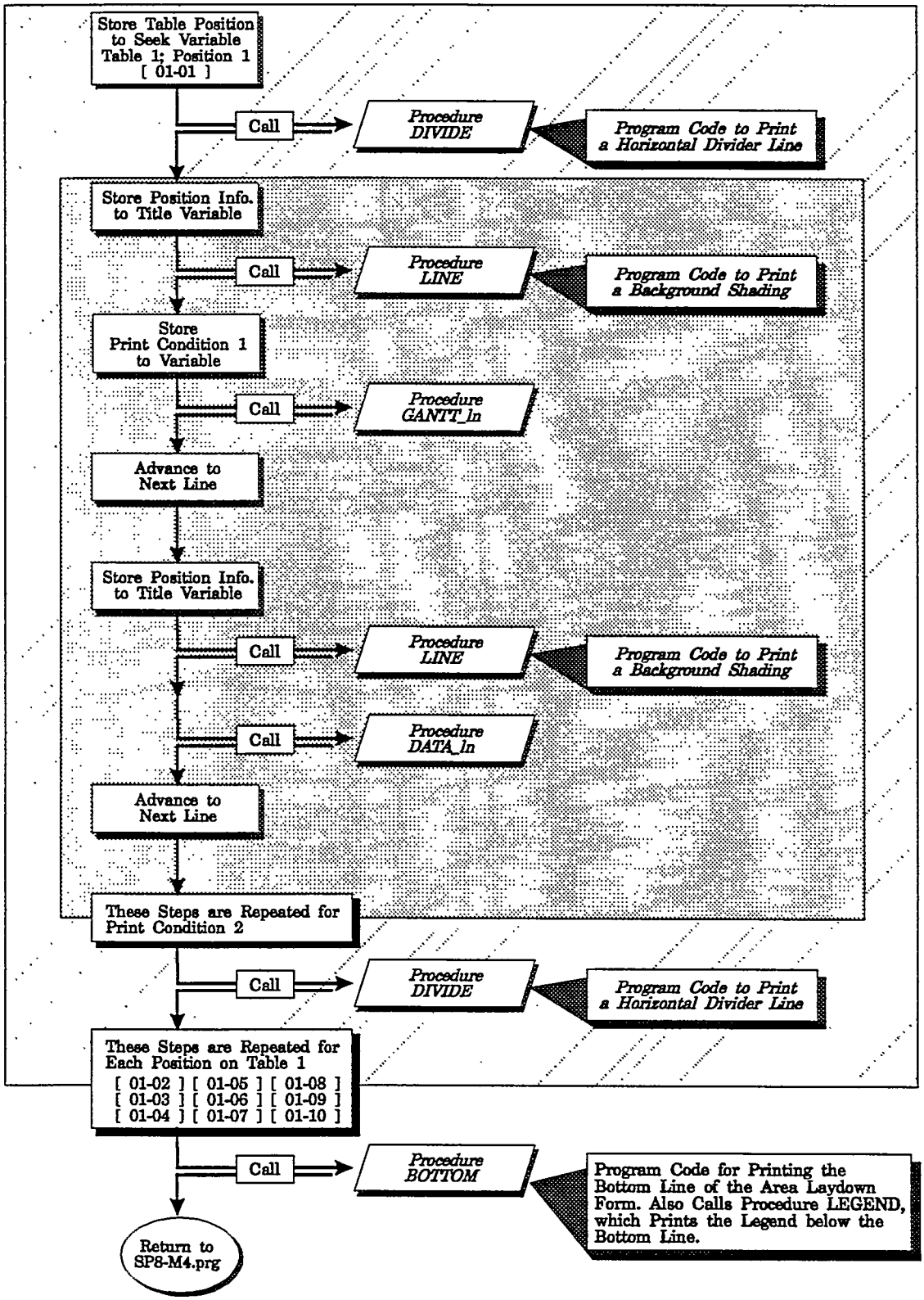
LOGIC DIAGRAM



PROCEDURE TABLE_1

LOGIC DIAGRAM

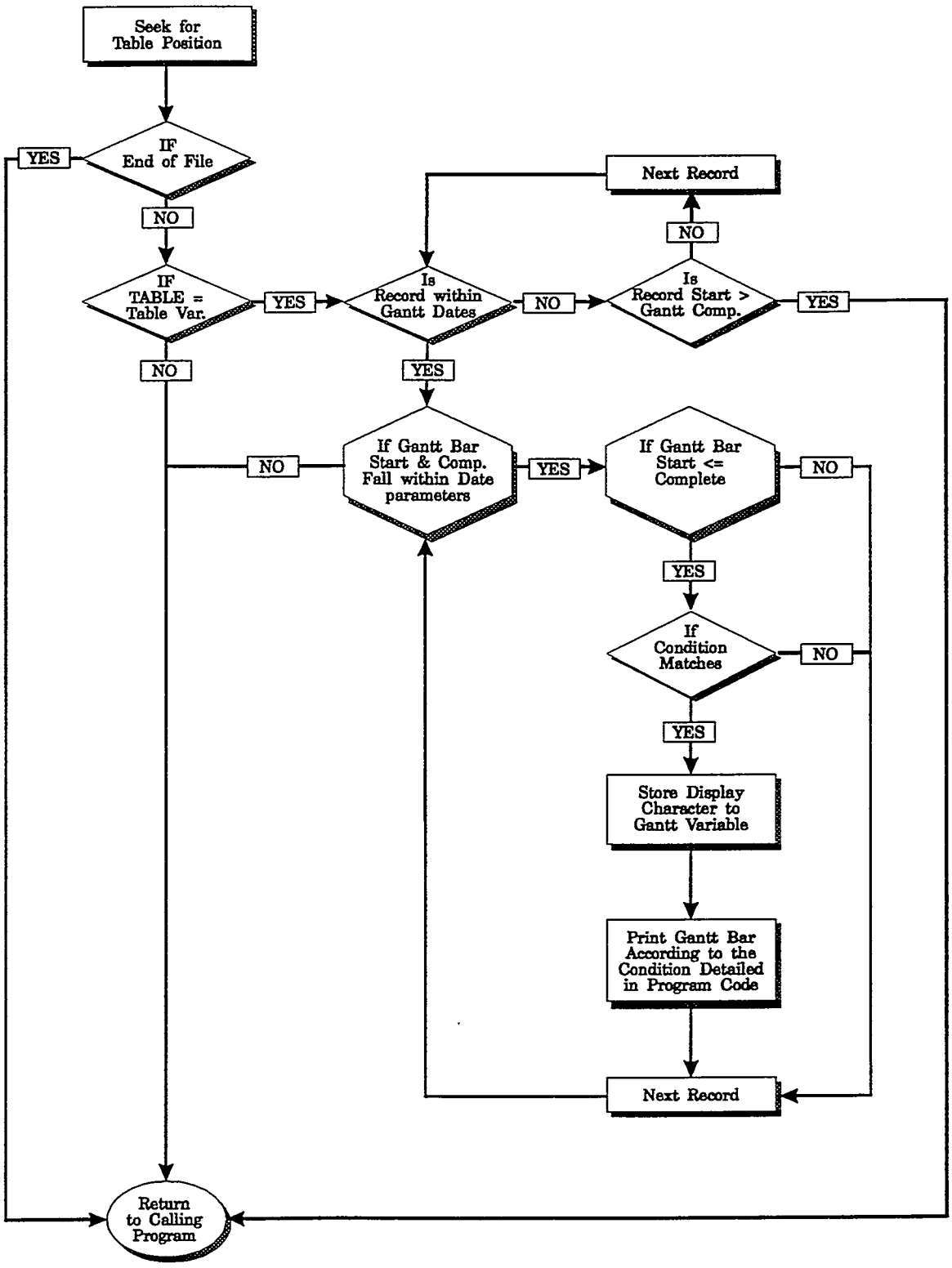
Procedures SHOP, TABLE_2, AREA_A & AREA_B are Similar



PROCEDURE GANTT_ln

LOGIC DIAGRAM

Procedure DATA_ln is similar



APPENDIX V

PROGRAM CODE

```
INPUT FILE:  C:\OPLAN\SP8_MENU.PRG
    1     *  FILE NAME: SP8_MENU.PRG
    2     *  BY: D. McQuaide
    3     *  DATE: March 24, 1992
    4     *  DESC:
    5     *  CALLED BY:
    6     *  DATA FILES:
    7     * SP8_MENU.prg
    8
    9     close all
   10     set talk   off
   11     set status off
   12     set safety off
   13     set color to gr/ ,w/r,
   14     clear
   15     set procedure to SP8_MENU
   16
   17     ********************************************************************************
   18 ┌──do while .T.
   19 │       set color to gr
   20 │       clear
   21 │
   22 │       store ' ' to Choice
   23 │       do MENU_1
   24 │       @ 15, 38  get Choice picture '!'
   25 │       read
   26 │    ┌──if readkey() = 12
   27 │    │       set color to w/b
   28 <===<===<===return
   29 │    └──endif
   30 │
   31 │       do MENU_1A
   32 │    ┌──do case
   33 │    ├──case Choice = '1'
   34 │    │       set procedure to
   35 │    │       do SP8_M1
   36 │    │       set procedure to SP8_MENU
   37 │    │
   38 │    ├──case Choice = '2'
   39 │    │       set procedure to
   40 │    │       do SP8_M2
   41 │    │       set procedure to SP8_MENU
   42 │    │
   43 │    ├──case Choice = '3'
   44 │    │       set procedure to
   45 │    │       do SP8_M3
   46 │    │       set procedure to SP8_MENU
   47 │    │
   48 │    ├──case Choice = '4'
   49 │    │       set procedure to
   50 │    │       do SP8_M4
   51 │    │       set procedure to SP8_MENU
   52 │    │
   53 │    └──endcase
   54 └──enddo
   55     ********************************************************************************
   56 <===return
   57
   58
   59     *********************************
   60     *    PROCEDURE MENU_1.prg       *
   61     *     Text for Main Menu        *
   62     *********************************
   63
   64     PROC MENU_1
   65
   66     set color to gr+/b
   67     @  2, 13  to  16, 54 double
   68     @  2, 25  SAY " INTEGRATED PLANNING "
   69     set color to w+/r,w/r
   70     @  3, 14  SAY "                                    "
   71     @  4, 14  SAY "     1: NEW MODEL GENERATION        "
   72     @  5, 14  SAY "                                    "
   73     @  6, 14  SAY "     2: PROJECT ANALYSIS            "
```

Read Main Menu Selection

```
   74     @  7, 14  SAY "                                          "
   75     @  8, 14  SAY "     3: BASELINE SCHEDULE GENERATION      "
   76     @  9, 14  SAY "                                          "
   77     @ 10, 14  SAY "     4: PRODUCTION SCHEDULE UPDATE        "
   78     @ 11, 14  SAY "                                          "
   79     @ 12, 14  SAY "                                          "
   80     @ 13, 14  SAY "                                          "
   81     @ 14, 14  SAY "                                          "
   82     @ 15, 14  SAY "          Enter Choice : :                "
   83     set color to gr
   84     @ 17, 23  say 'Press "Esc" to return'
   85
   86 <===return
   87
   88
   89
   90     *********************************
   91     *    PROCEDURE MENU_1A.prg      *
   92     *     Text for Main Menu        *
   93     * Shaded to allow for Overlay   *
   94     *********************************
   95
   96     PROC MENU_1A
   97
   98     set color to    /b
   99     @  2, 13  to  16, 54 double
  100     @  2, 25  SAY " INTEGRATED PLANNING "
  101     set color to w/bg
  102     @  3, 14  SAY "                                          "
  103     @  4, 14  SAY "     1: NEW MODEL GENERATION              "
  104     @  5, 14  SAY "                                          "
  105     @  6, 14  SAY "     2: PROJECT INTEGRATION               "
  106     @  7, 14  SAY "                                          "
  107     @  8, 14  SAY "     3: BASELINE SCHEDULE GENERATION      "
  108     @  9, 14  SAY "                                          "
  109     @ 10, 14  SAY "     4: PRODUCTION SCHEDULE UPDATE        "
  110     @ 11, 14  SAY "                                          "
  111     @ 12, 14  SAY "                                          "
  112     @ 13, 14  SAY "                                          "
  113     @ 14, 14  SAY "                                          "
  114     @ 15, 14  SAY "          Enter Choice : :                "
  115     set color to gr
  116     @ 17, 23  say 'Press "Esc" to return'
  117
  118 <===return
  119
  120     *Formatted by: dANALYST Ver. 7.3a on March 24, 1992 at 9:45 AM.
dANALYST found 0 error(s), 0 warning(s), 120 lines.
```

INPUT FILE: C:\OPLAN\SP8_M1.PRG

```
    1     *  FILE NAME: SP8_M1.PRG
    2     *  BY: D. McQuaide
    3     *  DATE: March 24, 1992
    4     *  DESC:
    5     *  CALLED BY:
    6     *  DATA FILES:
    7     * SP8_M1.prg
    8
    9     set procedure to SP8_M1
   10
   11     ********************************************************************************
   12  ┌──do while .T.
   13  │      set color to gr
   14  │      @  3, 15 clear
   15  │      store ' ' to ChoiceM1
   16  │      do MENU_1
   17  │      @ 16, 40  get ChoiceM1 picture '!'
   18  │      read
   19  │   ┌──if readkey() = 12
   20  <===<===<===return
   21  │   └──endif
   22  │
   23  │      do MENU_1A
   24  │   ┌──do case
   25  │   ├──case ChoiceM1 = '1'
   26  │   │      set procedure to
   27  │   │      do MODEL_B
   28  │   │      close all
   29  │   │      set procedure to SP8_M1
   30  │   │
   31  │   └──endcase
   32  └──enddo
   33     ********************************************************************************
   34  <===return
   35
   36
   37     ****************************************
   38     *       PROCEDURE MENU_1.prg           *
   39     * Text for New Model Generation Menu   *
   40     ****************************************
   41
   42     PROC MENU_1
   43
   44     set color to gr+/b
   45     @  3, 15  to  17, 56 double
   46     @  3, 26  SAY " NEW MODEL GENERATION "
   47     set color to w+/r
   48     @  4, 16  SAY "                                        "
   49     @  5, 16  SAY "     1: MODEL BUILDER PROGRAM           "
   50     @  6, 16  SAY "                                        "
   51     @  7, 16  SAY "                                        "
   52     @  8, 16  SAY "                                        "
   53     @  9, 16  SAY "                                        "
   54     @ 10, 16  SAY "                                        "
   55     @ 11, 16  SAY "                                        "
   56     @ 12, 16  SAY "                                        "
   57     @ 13, 16  SAY "                                        "
   58     @ 14, 16  SAY "                                        "
   59     @ 15, 16  SAY "                                        "
   60     @ 16, 16  SAY "          Enter Choice : :              "
   61     set color to gr
   62     @ 18, 25  say 'Press "Esc" to return'
   63
   64
   65  <===return
   66
   67
   68
   69     ****************************************
   70     *       PROCEDURE MENU_1A.prg          *
   71     * Text for New Model Generation Menu   *
   72     *    Shaded to allow for Overlay       *
   73     ****************************************
```

read New Model Generation Menu selection

```
   74
   75     PROC MENU_1A
   76
   77     set color to /b
   78     @  3, 15  to  17, 56 double
   79     @  3, 26  SAY " NEW MODEL GENERATION "
   80     set color to w/bg
   81     @  4, 16  SAY "                                          "
   82     @  5, 16  SAY "      1: MODEL BUILDER PROGRAM            "
   83     @  6, 16  SAY "                                          "
   84     @  7, 16  SAY "                                          "
   85     @  8, 16  SAY "                                          "
   86     @  9, 16  SAY "                                          "
   87     @ 10, 16  SAY "                                          "
   88     @ 11, 16  SAY "                                          "
   89     @ 12, 16  SAY "                                          "
   90     @ 13, 16  SAY "                                          "
   91     @ 14, 16  SAY "                                          "
   92     @ 15, 16  SAY "                                          "
   93     @ 16, 16  SAY "          Enter Choice : :                "
   94     set color to gr
   95     @ 18, 25  say 'Press "Esc" to return'
   96
   97
   98 <===return
   99
  100
  101
  102
  103
  104
  105
  106     *Formatted by: dANALYST Ver. 7.3a on March 24, 1992 at 9:45 AM.
dANALYST found 0 error(s), 0 warning(s), 106 lines.
```

INPUT FILE: C:\OPLAN\SP8_M2.PRG

```
    1     *  FILE NAME: SP8_M2.PRG
    2     *  BY: D. McQuaide
    3     *  DATE: March 24, 1992
    4     *  DESC:
    5     *  CALLED BY:
    6     *  DATA FILES:
    7     * SP8_M2.prg
    8
    9     set procedure to SP8_M2
   10
   11     ********************************************************************************
   12  ┌──do while .T.
   13  │      set color to gr
   14  │      @  3, 15 clear
   15  │      store ' ' to ChoiceM2
   16  │      do MENU_1
   17  │      @ 16, 40  get ChoiceM2 picture '!'
   18  │      read
   19  │   ┌──if readkey() = 12
   20  <===<===<===return
   21  │   └──endif
   22  │
   23  │      do MENU_1A
   24  │   ┌──do case
   25  │   ├──case ChoiceM2 = '1'
   26  │   │       set procedure to
   27  │   │       do SP8_DATA
   28  │   │       close all
   29  │   │       set procedure to SP8_M2
   30  │   │
   31  │   │
   32  │   └──endcase
   33  └──enddo
   34     ********************************************************************************
   35  <===return
   36
   37
   38     ************************************
   39     *      PROCEDURE MENU_1.prg        *
   40     *  Text for Project Analysis Menu  *
   41     ************************************
   42
   43     PROC MENU_1
   44
   45     set color to gr+/b
   46     @  3, 15  to  17, 56 double
   47     @  3, 28  SAY " PROJECT ANALYSIS "
   48     set color to w+/r
   49     @  4, 16  SAY "                                      "
   50     @  5, 16  SAY "     1: DATA GENERATION PROGRAM       "
   51     @  6, 16  SAY "        * Capacity Requirements.      "
   52     @  7, 16  SAY "        * Long Range Manning          "
   53     @  8, 16  SAY "          Requirements.               "
   54     @  9, 16  SAY "                                      "
   55     @ 10, 16  SAY "                                      "
   56     @ 11, 16  SAY "                                      "
   57     @ 12, 16  SAY "                                      "
   58     @ 13, 16  SAY "                                      "
   59     @ 14, 16  SAY "                                      "
   60     @ 15, 16  SAY "                                      "
   61     @ 16, 16  SAY "          Enter Choice : :            "
   62     set color to gr
   63     @ 18, 25  say 'Press "Esc" to return'
   64
   65
   66  <===return
   67
   68
   69
   70     ************************************
   71     *      PROCEDURE MENU_1A.prg       *
   72     *  Text for Project Analysis Menu  *
   73     *   Shaded to allow for Overlay    *
```

read Project Analysis Menu selection

```
   74     ************************************
   75
   76     PROC MENU_1A
   77
   78     set color to /b
   79     @  3, 15  to  17, 56 double
   80     @  3, 28  SAY " PROJECT ANALYSIS "
   81     set color to w/bg
   82     @  4, 16  SAY "                                      "
   83     @  5, 16  SAY "     1: DATA GENERATION PROGRAM       "
   84     @  6, 16  SAY "        * Capacity Requirements.      "
   85     @  7, 16  SAY "        * Long Range Manning          "
   86     @  8, 16  SAY "          Requirements.               "
   87     @  9, 16  SAY "                                      "
   88     @ 10, 16  SAY "                                      "
   89     @ 11, 16  SAY "                                      "
   90     @ 12, 16  SAY "                                      "
   91     @ 13, 16  SAY "                                      "
   92     @ 14, 16  SAY "                                      "
   93     @ 15, 16  SAY "                                      "
   94     @ 16, 16  SAY "          Enter Choice : :            "
   95     set color to gr
   96     @ 18, 25  say 'Press "Esc" to return'
   97
   98
   99 <===return
  100
  101
  102
  103
  104
  105
  106     *Formatted by: dANALYST Ver. 7.3a on March 24, 1992 at 9:45 AM.
dANALYST found 0 error(s), 0 warning(s), 106 lines.
```

```
INPUT FILE:  C:\OPLAN\SP8_M3.PRG
    1     *  FILE NAME: SP8_M3.PRG
    2     *  BY: D. McQuaide
    3     *  DATE: March 24, 1992
    4     *  DESC:
    5     *  CALLED BY:
    6     *  DATA FILES:
    7     * SP8_M3.prg
    8
    9     set procedure to SP8_M3
   10
   11     *************************************************************************
   12  ┌──do while .T.
   13  │      set color to gr
   14  │      @  3, 15 clear
   15  │      store ' ' to ChoiceM3
   16  │      do MENU_1
   17  │      @ 16, 40  get ChoiceM3 picture '!'
   18  │      read
   19  │   ┌──if readkey() = 12
   20  <===<===<===return
   21  │   └──endif
   22  │
   23  │      do MENU_1A
   24  │   ┌──do case
   25  │   ├──case ChoiceM3 = '1'
   26  │   │      set procedure to
   27  │   │      set color to gr+/b
   28  │   │      @  4,17 to 19,73 double
   29  │   │      @  4,34 say ' MASTER SCHEDULES MENU '
   30  │   │      set color to gr+/r
   31  │   │      @  5,18 say '                                                    '
   32  │   │      @  6,18 say '   MASTER SCHEDULES:                               '
   33  │   │      @  7,18 say '                                                    '
   34  │   │      set color to w+/r
   35  │   │      @  8,18 say ' The Master Schedule Generation Program uses the    '
   36  │   │      @  9,18 say ' activity file of the Baseline Production Model (after '
   37  │   │      @ 10,18 say ' processing by the project management software).  The '
   38  │   │      @ 11,18 say ' program extracts the dates necessary to generate the '
   39  │   │      @ 12,18 say ' various master schedules used throughout the yard.  '
   40  │   │      @ 13,18 say ' This module of the Integrated Production Planning   '
   41  │   │      @ 14,18 say ' System is dependent upon the format by which a      '
   42  │   │      @ 15,18 say ' specific yard produces its Master Production Schedules'
   43  │   │      @ 16,18 say ' Therefore, the details of this system module have   '
   44  │   │      @ 17,18 say ' not been developed for this presentation.          '
   45  │   │      @ 18,18 say '                                                    '
   46  │   │      set color to gr
   47  │   │      @ 20,33 say ' '
   48  │   │      wait
   49  │   │      @ 20,0  clear
   50  │   │      set procedure to SP8_M3
   51  │   │
   52  │   ├──case ChoiceM3 = '2'
   53  │   │      set procedure to
   54  │   │      do SP8_GNT
   55  │   │      close all
   56  │   │      set procedure to SP8_M3
   57  │   │
   58  │   ├──case ChoiceM3 = '3'
   59  │   │      set procedure to
   60  │   │      set color to gr+/b
   61  │   │      @  4,17 to 19,73 double
   62  │   │      @  4,35 say ' SCHEDULE UPLOAD MENU '
   63  │   │      set color to gr+/r
   64  │   │      @  5,18 say '                                                    '
   65  │   │      @  6,18 say '   SCHEDULE UPLOAD FILE:                           '
   66  │   │      @  7,18 say '                                                    '
   67  │   │      set color to w+/r
   68  │   │      @  8,18 say ' The Master Schedule Upload File Generation Program '
   69  │   │      @  9,18 say ' converts the Integrated Production Planning System '
   70  │   │      @ 10,18 say ' data to a form so that it may be uploaded into the '
   71  │   │      @ 11,18 say ' yards Cost Schedule Control System (CSCS) database. '
   72  │   │      @ 12,18 say ' This allows the CSCS to be rapidly and accurately  '
   73  │   │      @ 13,18 say ' updated when a change occurs to a master schedule. '
```

read Base Line Generation Menu selection

```
74          @ 14,18 say ' This module of the Integrated Production Planning     '
75          @ 15,18 say ' System is dependent upon the specifics of the yards   '
76          @ 16,18 say ' CSCS.  Therefore, the details of this system module   '
77          @ 17,18 say ' have not been developed for this presentation.        '
78          @ 18,18 say '                                                       '
79          set color to gr
80          @ 20,33 say ' '
81          wait
82          @ 20,0  clear
83          set procedure to SP8_M3
84
85      endcase
86  enddo
87  ********************************************************************************
88 <===return
89
90
91  ****************************************
92  *       PROCEDURE MENU_1.prg           *
93  * Text for Base Line Generation Menu   *
94  ****************************************
95
96  PROC MENU_1
97
98  set color to gr+/b
99  @  3, 15  to  17, 56 double
100 @  3, 26  SAY " BASE LINE GENERATION "
101 set color to w+/r
102 @  4, 16  SAY "                                    "
103 @  5, 16  SAY "     1: MASTER SCHEDULES            "
104 @  6, 16  SAY "         (Tabular Form)             "
105 @  7, 16  SAY "                                    "
106 @  8, 16  SAY "     2: MASTER SCHEDULES            "
107 @  9, 16  SAY "         (Gantt Chart Form)         "
108 @ 10, 16  SAY "                                    "
109 @ 11, 16  SAY "     3: SCHEDULE UPLOAD PROGRAM     "
110 @ 12, 16  SAY "                                    "
111 @ 13, 16  SAY "                                    "
112 @ 14, 16  SAY "                                    "
113 @ 15, 16  SAY "                                    "
114 @ 16, 16  SAY "          Enter Choice : :          "
115 set color to gr
116 @ 18, 25  say 'Press "Esc" to return'
117
118
119 <===return
120
121
122
123 ****************************************
124 *       PROCEDURE MENU_1A.prg          *
125 * Text for Base Line Generation Menu   *
126 *    Shaded to allow for Overlay       *
127 ****************************************
128
129 PROC MENU_1A
130
131 set color to /b
132 @  3, 15  to  17, 56 double
133 @  3, 26  SAY " BASE LINE GENERATION "
134 set color to w/bg
135 @  4, 16  SAY "                                    "
136 @  5, 16  SAY "     1: MASTER SCHEDULES            "
137 @  6, 16  SAY "         (Tabular Form)             "
138 @  7, 16  SAY "                                    "
139 @  8, 16  SAY "     2: MASTER SCHEDULES            "
140 @  9, 16  SAY "         (Gantt Chart Form)         "
141 @ 10, 16  SAY "                                    "
142 @ 11, 16  SAY "     3: SCHEDULE UPLOAD PROGRAM     "
143 @ 12, 16  SAY "                                    "
144 @ 13, 16  SAY "                                    "
145 @ 14, 16  SAY "                                    "
146 @ 15, 16  SAY "                                    "
147 @ 16, 16  SAY "          Enter Choice : :          "
148 set color to gr
```

```
  149     @ 18, 25  say 'Press "Esc" to return'
  150
  151
  152 <===return
  153
  154
  155
  156
  157
  158
  159
  160
  161     *Formatted by: dANALYST Ver. 7.3a on March 24, 1992 at 9:46 AM.
```

dANALYST found 0 error(s), 0 warning(s), 161 lines.

```
INPUT FILE:  C:\OPLAN\SP8_M4.PRG
    1     *  FILE NAME: SP8_M4.PRG
    2     *  BY: D. McQuaide
    3     *  DATE: March 24, 1992
    4     *  DESC:
    5     *  CALLED BY:
    6     *  DATA FILES:
    7     * SP8_M4.prg
    8
    9     set procedure to SP8_M4
   10
   11     ********************************************************************************
   12  ┌──do while .T.
   13  │      set color to gr
   14  │      @  3, 15 clear
   15  │      store ' ' to ChoiceM4
   16  │      do MENU_1
   17  │      @ 16, 40  get ChoiceM4 picture '!'
   18  │      read
   19  │   ┌──if readkey() = 12
   20  <===<===<===return
   21  │   └──endif
   22  │
   23  │      do MENU_1A
   24  │   ┌──do case
   25  │   ├──case ChoiceM4 = '1'
   26  │   │      set procedure to
   27  │   │      set color to gr+/b
   28  │   │      @  4,17 to 18,73 double
   29  │   │      @  4,33 say ' PRODUCTION SCHEDULE MENU '
   30  │   │      set color to gr+/r
   31  │   │      @  5,18 say '                                                          '
   32  │   │      @  6,18 say '   PRODUCTION SCHEDULE:                                  '
   33  │   │      @  7,18 say '                                                          '
   34  │   │      set color to w+/r
   35  │   │      @  8,18 say ' The Production Schedule Generation Program uses the     '
   36  │   │      @  9,18 say ' activity file of the Production Update Model (after     '
   37  │   │      @ 10,18 say ' processing by the project management software). The     '
   38  │   │      @ 11,18 say ' program takes the activity file and extracts the        '
   39  │   │      @ 12,18 say ' dates necessary to generate the updataed production     '
   40  │   │      @ 13,18 say ' schedules in both graphical and tabular form.  These    '
   41  │   │      @ 14,18 say ' schedules show planned vs. actual and projected         '
   42  │   │      @ 15,18 say ' progress. Each Shipyard should create this module to    '
   43  │   │      @ 16,18 say ' meet thier own needs.                                    '
   44  │   │      @ 17,18 say '                                                          '
   45  │   │      set color to gr
   46  │   │      @ 19,33 say ' '
   47  │   │      wait
   48  │   │      @ 19,0  clear
   49  │   │      set procedure to SP8_M4
   50  │   │
   51  │   ├──case ChoiceM4 = '2'
   52  │   │      set procedure to
   53  │   │      set color to gr+/b
   54  │   │      @  4,17 to 18,73 double
   55  │   │      @  4,33 say ' SHORT TERM MANPOWER MENU '
   56  │   │      set color to gr+/r
   57  │   │      @  5,18 say '                                                          '
   58  │   │      @  6,18 say '   SHORT TERM MANPOWER:                                  '
   59  │   │      @  7,18 say '                                                          '
   60  │   │      set color to w+/r
   61  │   │      @  8,18 say ' The Short Term Manning Requirement Generation Program    '
   62  │   │      @  9,18 say ' is similar to the Long Range Manning Requirement         '
   63  │   │      @ 10,18 say ' Generation Program.  However, the short term             '
   64  │   │      @ 11,18 say ' requirements are generated from the resource             '
   65  │   │      @ 12,18 say ' aggregation file of the Production Update Model.         '
   66  │   │      @ 13,18 say ' The Long Range Manning Program can be modified to        '
   67  │   │      @ 14,18 say ' show short term projections.                             '
   68  │   │      @ 15,18 say '                                                          '
   69  │   │      @ 16,18 say '                                                          '
   70  │   │      @ 17,18 say '                                                          '
   71  │   │      set color to gr
   72  │   │      @ 19,33 say ' '
   73  │   │      wait
```

read Production Schedule Upd Menu selection

```
74  |   |       @ 19,0  clear
75  |   |       set procedure to SP8_M4
76  |   |
77  |   |---case ChoiceM4 = '3'
78  |   |       set procedure to
79  |   |       do SP8_LAY
80  |   |       close all
81  |   |       set procedure to SP8_M4
82  |   |
83  |   |---endcase
84  |---enddo
85      ********************************************************************************
86  <===return
87
88
89      ***********************************************
90      *          PROCEDURE MENU_1.prg               *
91      * Text for Production Schedule Update  Menu   *
92      ***********************************************
93
94      PROC MENU_1
95
96      set color to gr+/b
97      @  3, 15  to  17, 56 double
98      @  3, 23  SAY " PRODUCTION SCHEDULE UPDATE "
99      set color to w+/r
100     @  4, 16  SAY "                                    "
101     @  5, 16  SAY "     1: PRODUCTION SCHEDULES        "
102     @  6, 16  SAY "                                    "
103     @  7, 16  SAY "     2: SHORT TERM MANNING          "
104     @  8, 16  SAY "                                    "
105     @  9, 16  SAY "     3: LAYDOWN SCHEDULES           "
106     @ 10, 16  SAY "                                    "
107     @ 11, 16  SAY "                                    "
108     @ 12, 16  SAY "                                    "
109     @ 13, 16  SAY "                                    "
110     @ 14, 16  SAY "                                    "
111     @ 15, 16  SAY "                                    "
112     @ 16, 16  SAY "          Enter Choice : :          "
113     set color to gr
114     @ 18, 25  say 'Press "Esc" to return'
115
116
117 <===return
118
119
120
121     ***********************************************
122     *          PROCEDURE MENU_1A.prg              *
123     * Text for Production Schedule Update  Menu   *
124     *       Shaded to allow for Overlay           *
125     ***********************************************
126
127     PROC MENU_1A
128
129     set color to /b
130     @  3, 15  to  17, 56 double
131     @  3, 23  SAY " PRODUCTION SCHEDULE UPDATE "
132     set color to w/bg
133     @  4, 16  SAY "                                    "
134     @  5, 16  SAY "     1: PRODUCTION SCHEDULES        "
135     @  6, 16  SAY "                                    "
136     @  7, 16  SAY "     2: SHORT TERM MANNING          "
137     @  8, 16  SAY "                                    "
138     @  9, 16  SAY "     3: LAYDOWN SCHEDULES           "
139     @ 10, 16  SAY "                                    "
140     @ 11, 16  SAY "                                    "
141     @ 12, 16  SAY "                                    "
142     @ 13, 16  SAY "                                    "
143     @ 14, 16  SAY "                                    "
144     @ 15, 16  SAY "                                    "
145     @ 16, 16  SAY "          Enter Choice : :          "
146     set color to gr
147     @ 18, 25  say 'Press "Esc" to return'
148
```

```
  149
  150 <===return
  151
  152
  153
  154
  155
  156     *Formatted by: dANALYST Ver. 7.3a on March 24, 1992 at 9:46 AM.
dANALYST found 0 error(s), 0 warning(s), 156 lines.
```

INPUT FILE: C:\OPLAN\MODEL_B.PRG

```
    1  *********************************************************
    2  *       Program: MODEL_B.prg                            *
    3  * Programmed by: D.J.McQuaide & R.J.Neumann              *
    4  *     Developed: 07/22/91                               *
    5  *       Purpose:                                        *
    6  *                                                       *
    7  *                                                       *
    8  *                                                       *
    9  *********************************************************
   10  set procedure to MODEL_B
   11
   12
   13  * Input Menu *
   14
   15  ************************************************************************
   16  store ' '       to tContract
   17  store space(30)  to tProject
   18  store space(30) to tBuild_DB
   19  do MODEL_M
   20  @ 7,46 get tContract picture '!'
   21  do while .T.
   22      @ 9,23 get tProject  picture '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
   23      read
   24      if readkey() = 12
   25  <===<===<===return
   26      endif
   27      store ' ' to tLoop
   28      store trim(tProject)+'.act' to tACT
   29      store trim(tProject)+'.rel' to tREL
   30      store trim(tProject)+'.res' to tRES
   31      set color to gr+/r
   32      if .not. file(tACT)
   33          @ 14, 18  SAY '       Activity File NOT Found!        '
   34          store 'Y' to tLoop
   35      endif
   36      if .not. file(tREL)
   37          @ 15, 18  SAY '       Relation File NOT Found!        '
   38          store 'Y' to tLoop
   39      endif
   40      if .not. file(tRES)
   41          @ 16, 18  SAY '       Resource File NOT Found!        '
   42          store 'Y' to tLoop
   43      endif
   44      if tLoop = 'Y'
   45  <===<===<===loop
   46      else
   47  <===<===<===exit
   48      endif
   49  enddo
   50  ************************************************************************
   51  set color to w+/r
   52  @ 14, 18  SAY '                                               '
   53  @ 15, 18  SAY '                                               '
   54  @ 16, 18  SAY '                                               '
   55  do while .T.
   56      @ 12,23 get tBuild_DB picture '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
   57      read
   58      if readkey() = 12
   59  <===<===<===return
   60      endif
   61      store trim(tBuild_DB)+'.dbf' to tBUILD
   62      set color to gr+/r
   63      if .not. file(tBUILD)
   64          @ 14, 18  SAY '      Build Data File NOT Found!       '
   65  <===<===<===loop
   66      else
   67  <===<===<===exit
   68      endif
   69  enddo
   70  ************************************************************************
   71  set color to w+/r
   72  @ 14, 18  SAY '                                               '
   73
```

reading name of model and verifies all files exist

reading and verifying the BUILD data file

```
74      * Set-Up Data Bases *
75      set color to gr+*/r
76      @ 14, 25  SAY '*'
77      @ 14, 49  SAY '*'
78      set color to gr+/r
79      @ 14, 18  SAY '         Setting Up Data Bases       '
80
81
82      select D
83      use &tRES alias RES
84      zap
85
86      select C
87      use &tREL alias REL
88      zap
89
90      select B
91      use &tACT alias ACT
92      zap
93
94      select A
95      use &tBUILD index GRD_BLK alias BUILD
96      reindex
97
98      set color to gr+/r
99      @ 14, 18  SAY '                                     '
100
101     ************************************************************************
102     * Start Model Building Process *
103     go top
104   --do while .not. eof()
105         set color to gr+/r
106         @ 14, 18  SAY '  Add Block                      '
107         @ 14, 30  SAY BUILD->Block
108         @ 15, 18  SAY '                Activity Records       '
109         @ 16, 18  SAY '                Relation Records       '
110         @ 17, 18  SAY '                Resource Records       '
111         ************************************************************************
112       --do case
113       --case BUILD->STRATEGY = 'STD'
114
115           set color to gr+/r
116           @ 14, 35  SAY 'Standard Strategy      '
117           @ 14, 30  SAY BUILD->Block
118           * Adding ACTIVITY Records *
119           do ACT1
120           * Adding RELATION Records *
121           do REL1
122           * Adding RESOURCE Records *
123           do RES1
124
125       --case BUILD->STRATEGY = 'GRAND1'
126           *** Grand Blocking (Post Blast) ***
127           set color to gr+/r
128           @ 14, 35  SAY 'Grand Block #1 Strat.  '
129
130           select BUILD
131           store recno() to tRecord
132           store BUILD->GRAND to tGrand
133         --do while BUILD->GRAND = tGrand .and. BUILD->BLOCK < '500'
134               set color to gr+/r
135               @ 14, 30  SAY BUILD->Block
136               * Adding ACTIVITY Records *
137               do ACT1
138               * Adding RELATION Records *
139               do REL1
140               * Adding RESOURCE Records *
141               do RES1
142
143               select BUILD
144               skip
145               @ 15, 30  SAY '    '
146               @ 16, 30  SAY '    '
147               @ 17, 30  SAY '    '
148           --enddo
```

set up required data bases

displays Model Build Prog process to screen

```
224     @ 7, 33  say trim(tProject)
225     store ' ' to tP_Lanes
226     set color to gr+/r
227     @ 8, 18  SAY '                                          '
228     @ 9, 18  SAY '     Has successfully been created.       '
229     @ 10, 18  SAY '                                         '
230     set color to w+/r
231     @ 11, 18  SAY '     Do you want to add any Process      '
232     @ 12, 18  SAY '     Lane relationship via a File.       '
233     @ 13, 18  SAY '     (Y/N) : :                           '
234     @ 14, 18  SAY '                                         '
235     @ 15, 18  SAY '                                         '
236     @ 16, 18  SAY '                                         '
237     @ 17, 18  SAY '                                         '
238     @ 13, 30  Get tP_Lanes picture '!'
239     read
240 ┌---if readkey() = 12
241 <===<===return
242 └---endif
243     ********************************************************************************
244 ┌---if tP_Lanes = 'Y'
245 |       set color to w+/r
246 |       @ 9, 18  SAY '                                    '
247 |       @ 11, 18  SAY '                                   '
248 |       @ 12, 18  SAY '                                   '
249 |       @ 13, 18  SAY "  Enter Process Lane's File Name w/Path "
250 |   ┌---do while .T.
251 |   |       @ 14,23 get tP_Lane picture '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
252 |   |       read
253 |   |   ┌---if readkey() = 12
254 <===<===<===<===return
255 |   |   └---endif
256 |   |       store trim(tP_Lane)+'.dbf' to dbP_LANE
257 |   |   ┌---if file(dbP_LANE)
258 |   |   |       set color to w+/r
259 |   |   |       select REL
260 |   |   |       @ 16, 18  SAY '     Adding Process Lane Relations      '
261 |   |   |       append from &dbP_LANE
262 |   |   |       @ 16, 18  SAY '                                        '
263 |   <===<===<===exit
264 |   |   ├---else
265 |   |   |       set color to w+/r
266 |   |   |       @ 16, 18  SAY '           Can Not Find File            '
267 |   <===<===<===loop
268 |   |   └---endif
269 |   └---enddo
270 |       set color to w+/r
271 |       @ 16, 18  SAY '                                    '
272 └---endif
273     ********************************************************************************
274     store ' ' to tCodes
275     set color to w+/r
276     @ 7, 18  say '                                          '
277     @ 8, 18  say '    Do you to Create New Code Files       '
278     @ 9, 18  say '            ( es/ o) : :                  '
279     @ 10, 18  say '                                         '
280     @ 11, 18  say '                                         '
281     @ 12, 18  say '                                         '
282     @ 13, 18  say '                                         '
283     @ 14, 18  say '                                         '
284     set color to gr+/r
285     @ 9, 31  say 'Y'
286     @ 9, 35  say 'N'
287     @ 9, 40  get tCodes picture '!'
288     read
289     ********************************************************************************
290 ┌---if tCodes = 'Y'
291 |       set color to w+/r, /w
292 |       @ 12, 18  say '    Setting Up Indexes                 '
293 |       set unique on
294 |       select ACT
295 |       @ 13, 18  say ' C1 for Code Field #1 (WBS)            '
296 |       index on C1 to C1
297 |       @ 13, 18  say ' C4 for Code Field #4 (Grand Blk Code) '
298 |       index on C4 to C4
```

displays that project has been created and asks if you wish to add process lanes to the model

gets PROCESS LANE Data File and creates process lane relationships

asks if you wish to add code fields to the model and sets up data files

```
149         set color to gr+/r
150         @ 14, 30  SAY BUILD->Block
151         * Adding ACTIVITY Records *
152         do ACTG1
153         * Adding RELATION Records *
154         do RELG1
155         * Adding RESOURCE Records *
156         do RESG1
157
158     case BUILD->STRATEGY = 'GRAND2' .or. BUILD->STRATEGY = 'GRAND3' .or. BUILD->STRATEGY = 'GRAND4'
159         *** Grand Blocking (ON - ASSY) ***
160         set color to gr+/r
161         @ 14, 35  SAY 'Grand Block #2 Strat.  '
162         @ 14, 30  SAY BUILD->Block
163
164         select BUILD
165         store recno() to tRecord
166         store BUILD->GRAND to tGrand
167         do while BUILD->GRAND = tGrand .and. BUILD->BLOCK < '500'
168             set color to gr+/r
169             @ 14, 30  SAY BUILD->Block
170             * Adding ACTIVITY Records *
171             do ACT2
172             * Adding RELATION Records *
173             do REL2
174             * Adding RESOURCE Records *
175             do RES2
176
177             select BUILD
178             skip
179             @ 15, 30  SAY '    '
180             @ 16, 30  SAY '    '
181             @ 17, 30  SAY '    '
182         enddo
183         set color to gr+/r
184         @ 14, 30  SAY BUILD->Block
185         * Adding ACTIVITY Records *
186         store '    ' to tStack_of
187         do ACTG2
188         * Adding RELATION Records *
189         do RELG2
190         * Adding RESOURCE Records *
191         do RESG2
192
193     case BUILD->STRATEGY = 'SHOP'
194
195         set color to gr+/r
196         @ 14, 35  SAY 'SHOP Strategy          '
197         @ 14, 30  SAY BUILD->Block
198         * Adding ACTIVITY Records *
199         do ACT1
200         * Adding RELATION Records *
201         do REL3
202         * Adding RESOURCE Records *
203         do RES1
204
205         set color to gr+/r
206         @ 14, 35  SAY 'Shop Block Strategy    '
207         @ 14, 30  SAY BUILD->Block
208
209     endcase
210     set color to w+/r
211     @ 15, 30  SAY '    '
212     @ 16, 30  SAY '    '
213     @ 17, 30  SAY '    '
214     select BUILD
215     skip
216 enddo
217 ************************************************************************
218 store space(30) to tP_Lane
219 set color to w+/r
220 @  5, 18  SAY '                                        '
221 @  6, 18  SAY '                                        '
222 @  7, 18  SAY '     Project :                          '
223 set color to gr+/r
```

Creating Activity, Resource, and Relationship files according to Block Build Strategy

```
299         @ 13, 18  say '                                          '
300         store 'Y' to tC1,tC4
301         store space(30) to fCode
302         @ 12, 18  say '    Varify Re-creating Code #1 : :      '
303         @ 12, 50  get tC1 picture '!'
304         read
305      ---if tC1 = 'Y'
306         |   store 'Y' to tContinue
307         |   @ 13, 18  say '      Enter Code File w/Path           '
308         |---do while .T.
309         |   |   @ 14, 24  get fCode picture '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
310         |   |   read
311         |   |---if readkey() = 12
312         |   |   |   store 'N' to tContinue
313         |   <===<===<===exit
314         |   |---endif
315         |   |   store trim(fCode)+'.COD' to tCode_File
316         |   |---if file(tCode_File)
317         |   |   |   @ 12, 18  say '                                '
318         |   |   |   @ 13, 18  say '                                '
319         |   |   |   @ 14, 18  say '                                '
320         |   |   |   @ 15, 18  say '                                '
321         |   <===<===<===exit
322         |   |   |---else
323         |   |   |   @ 15, 18  say '        Can Not Find File       '
324         |   <===<===<===loop
325         |   |---endif
326         |---enddo
327         |---if tContinue = 'Y'
328         |   |   set color to w+/r, /w
329         |   |   @ 12, 18  say '    Creating New Code 1 File        '
330         |   |   @ 13, 24  say tCode_File
331         |   |   select ACT
332         |   |   set index to C1
333         |   |   select E
334         |   |   use &tCode_File alias Code
335         |   |   zap
336         |   |   append blank
337         |   |   replace C1 with '1'
338         |   |   replace C1DESC with 'Production'
339         |   |   append blank
340         |   |   replace C1 with '1'+tContract
341         |   |   replace C1DESC with 'Contract "'+tContract+'"'
342         |   |
343         |   |   select ACT
344         |   |   go top
345         |   |---do while .not. eof()
346         |   |   |   store substr(C1,3,4) to tBlock
347         |   |   |   @ 15, 18  say '    For Block                   '
348         |   |   |   @ 15, 32  say tBlock
349         |   |   |   select CODE
350         |   |   |   append blank
351         |   |   |   replace C1 with '1'+tContract+tBlock
352         |   |   |   replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock
353         |   |   |   select ACT
354         |   |   |---do while substr(C1,3,4) = tBlock .and. .not. eof()
355         |   |   |   |---if substr(C1,7,1) = '1'
356         |   |   |   |   |   @ 15, 37  say 'Steel     '
357         |   |   |   |   |   select CODE
358         |   |   |   |   |   append blank
359         |   |   |   |   |   replace C1 with '1'+tContract+tBlock+'1'
360         |   |   |   |   |   replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel'
361         |   |   |   |   |   select ACT
362         |   |   |   |   |---do while substr(C1,7,1) = '1'
363         |   |   |   |   |   |---do case
364         |   |   |   |   |   |---case C1 = '1'+tContract+tBlock+'11'
365         |   |   |   |   |   |   |   select CODE
366         |   |   |   |   |   |   |   append blank
367         |   |   |   |   |   |   |   replace C1 with '1'+tContract+tBlock+'11'
368         |   |   |   |   |   |   |   replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Fab.'
369         |   |   |   |   |   |---case C1 = '1'+tContract+tBlock+'12'
370         |   |   |   |   |   |   |   select CODE
371         |   |   |   |   |   |   |   append blank
372         |   |   |   |   |   |   |   replace C1 with '1'+tContract+tBlock+'12'
373         |   |   |   |   |   |   |   replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Sub-Assy'
```

builds code field (WBS)

```
374                                  case C1 = '1'+tContract+tBlock+'131'
375                                       select CODE
376                                       append blank
377                                       replace C1 with '1'+tContract+tBlock+'13'
378                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Assembly'
379                                       append blank
380                                       replace C1 with '1'+tContract+tBlock+'131'
381                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Assy "Flat Block"
382                                  case C1 = '1'+tContract+tBlock+'132'
383                                       select CODE
384                                       append blank
385                                       replace C1 with '1'+tContract+tBlock+'13'
386                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Assembly'
387                                       append blank
388                                       replace C1 with '1'+tContract+tBlock+'132'
389                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Assy "Curved Bloc
390                                  case C1 = '1'+tContract+tBlock+'133'
391                                       select CODE
392                                       append blank
393                                       replace C1 with '1'+tContract+tBlock+'13'
394                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Assembly'
395                                       append blank
396                                       replace C1 with '1'+tContract+tBlock+'133'
397                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Assy "Tilt Jig"'
398                                  case C1 = '1'+tContract+tBlock+'141'
399                                       select CODE
400                                       append blank
401                                       replace C1 with '1'+tContract+tBlock+'14'
402                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Erection'
403                                       append blank
404                                       replace C1 with '1'+tContract+tBlock+'141'
405                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Erection "On-Boar
406                                  case C1 = '1'+tContract+tBlock+'142'
407                                       select CODE
408                                       append blank
409                                       replace C1 with '1'+tContract+tBlock+'14'
410                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Erection'
411                                       append blank
412                                       replace C1 with '1'+tContract+tBlock+'142'
413                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Steel Erection "Grand B
414                                  endcase
415                                  select ACT
416                                  skip
417                             enddo
418                        endif
419                        if substr(C1,7,1) = '2'
420                             @ 15, 37  say 'Non-Steel '
421                             select CODE
422                             append blank
423                             replace C1 with '1'+tContract+tBlock+'2'
424                             replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Non-Steel'
425                             select ACT
426                             do while substr(C1,7,1) = '2'
427                                  do case
428                                  case C1 = '1'+tContract+tBlock+'21'
429                                       select CODE
430                                       append blank
431                                       replace C1 with '1'+tContract+tBlock+'21'
432                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Pipe Shop'
433                                  case C1 = '1'+tContract+tBlock+'22'
434                                       select CODE
435                                       append blank
436                                       replace C1 with '1'+tContract+tBlock+'22'
437                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Vent Shop'
438                                  case C1 = '1'+tContract+tBlock+'2311'
439                                       select CODE
440                                       append blank
441                                       replace C1 with '1'+tContract+tBlock+'23'
442                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' On-Block O/F'
443                                       append blank
444                                       replace C1 with '1'+tContract+tBlock+'231'
445                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Pre-Blast O/F'
446                                       append blank
447                                       replace C1 with '1'+tContract+tBlock+'2311'
448                                       replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Pre-Blast O/F "Inverted'
```

```
449                     case C1 = '1'+tContract+tBlock+'2312'
450                         select CODE
451                         append blank
452                         replace C1 with '1'+tContract+tBlock+'23'
453                         replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' On-Block O/F'
454                         append blank
455                         replace C1 with '1'+tContract+tBlock+'231'
456                         replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Pre-Blast O/F'
457                         append blank
458                         replace C1 with '1'+tContract+tBlock+'2312'
459                         replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Pre-Blast O/F "Up-Right"'
460                     case C1 = '1'+tContract+tBlock+'232'
461                         select CODE
462                         append blank
463                         replace C1 with '1'+tContract+tBlock+'23'
464                         replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' On-Block O/F'
465                         append blank
466                         replace C1 with '1'+tContract+tBlock+'232'
467                         replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Post-Blast O/F'
468                     case C1 = '1'+tContract+tBlock+'24'
469                         select CODE
470                         append blank
471                         replace C1 with '1'+tContract+tBlock+'24'
472                         replace C1DESC with 'Contract "'+tContract+'" Block '+tBlock+' Blast & Paint'
473                     endcase
474                     select ACT
475                     skip
476                 enddo
477                 endif
478                 @ 15, 37  say '          '
479             enddo
480         enddo
481         @ 12, 18  say '                                        '
482         @ 13, 18  say '                                        '
483         @ 15, 18  say '                                        '
484     endif
485 endif
486 ************************************************************************
487 @ 12, 18  say '    Varify Re-creating Code #4 : :      '
488 @ 12, 50  get tC4 picture '!'
489 if tC4 = 'Y'
490     store space(30) to fCode
491     store 'Y' to tContinue
492     @ 13, 18  say '      Enter Code File w/Path          '
493     do while .T.
494         @ 14, 24  get fCode picture '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
495         read
496         if readkey() = 12
497             store 'N' to tContinue
498 <===<===<===exit
499         endif
500         store trim(fCode)+'.COD' to tCode_File
501         if file(tCode_File)
502             @ 12, 18  say '                                        '
503             @ 13, 18  say '                                        '
504             @ 14, 18  say '                                        '
505             @ 15, 18  say '                                        '
506 <===<===<===exit
507         else
508             @ 15, 18  say '        Can Not Find File               '
509 <===<===<===loop
510         endif
511     enddo
512     if tContinue = 'Y'
513         set color to w+/r, /w
514         @ 12, 18  say '    Creating Temp File from Activity    '
515         select ACT
516         set index to C4
517         copy to tempcode field C4
518         select E
519         @ 12, 18  say '    Creating New Code 4 File            '
520         @ 13, 24  say tCode_File
521         use &tCode_File alias Code
522         zap
523         append from tempcode
```

builds code field 4 (Grand Block Code)

```
524                 replace all C4DESC with 'Activity Related to Grand Block '+C4
525                 @ 12, 18  say '                                            '
526                 @ 13, 18  say '                                            '
527             endif
528         endif
529
530         set unique off
531         set color to w+/r,w/r
532     endif
533     ***************************************************************************
534
535
536 <===return
537
538
539     ****************************************************************
540     *  ACT1 ----> Creates Activity File Data for Blocks            *
541     *                                                              *
542     *                                                              *
543     ****************************************************************
544
545     PROC ACT1
546
547     * Adding ACTIVITY Records *
548     set color to w+*/r
549     @ 15, 30  SAY '--->'
550     set color to w+/r
551     select ACT
552     * Erection *
553     append blank
554     if BUILD->GRAND = '    '
555         replace ID with tContract+'7160'+BUILD->BLOCK
556         replace DS with 'Erection of Block '+BUILD->BLOCK
557         replace D with 20
558         replace TARGS with BUILD->ERECTION
559         replace TARGSTYPE with 'ON'
560         replace CAL with 1
561         replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'141'
562         replace C2 with '1141'
563         replace C3 with BUILD->BLK_TYPE
564         replace C4 with BUILD->GRAND
565     else
566         replace ID with tContract+'7150'+BUILD->BLOCK
567         replace DS with 'Stack of '+BUILD->BLOCK+' to '+BUILD->GRAND
568         replace D with 10
569         replace CAL with 1
570         replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'142'
571         replace C2 with '1142'
572         replace C3 with BUILD->BLK_TYPE
573         replace C4 with BUILD->GRAND
574     endif
575     * Post Blast O/F *
576     append blank
577     replace ID with tContract+'7254'+BUILD->BLOCK
578     replace DS with 'Post Blast O/F of Block '+BUILD->BLOCK
579     replace D with BUILD->OF4_D
580     replace CAL with 1
581     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'232'
582     replace C2 with '1232'
583     replace C3 with BUILD->BLK_TYPE
584     replace C4 with BUILD->GRAND
585     * Blast & Painting *
586     append blank
587     replace ID with tContract+'7253'+BUILD->BLOCK
588     replace DS with 'Blast & Painting of Block '+BUILD->BLOCK
589     replace D with BUILD->OF3_D
590     replace CAL with 1
591     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'24'
592     replace C2 with '124'
593     replace C3 with BUILD->BLK_TYPE
594     replace C4 with BUILD->GRAND
595     * Pre Blast (U-R) O/F *
596     append blank
597     replace ID with tContract+'7252'+BUILD->BLOCK
598     replace DS with 'Pre Blast (U-R) O/F of Block '+BUILD->BLOCK
```

```
599     replace D with BUILD->OF2_D
600     replace CAL with 1
601     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'2312'
602     replace C2 with '12312'
603     replace C3 with BUILD->BLK_TYPE
604     replace C4 with BUILD->GRAND
605     * Pre Blast (INV) O/F *
606     append blank
607     replace ID with tContract+'7251'+BUILD->BLOCK
608     replace DS with 'Pre Blast (INV) O/F of Block '+BUILD->BLOCK
609     replace D with BUILD->OF1_D
610     replace CAL with 1
611     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'2311'
612     replace C2 with '12311'
613     replace C3 with BUILD->BLK_TYPE
614     replace C4 with BUILD->GRAND
615     * Assembly *
616  ┌──if BUILD->STRATEGY = 'STD' .or. BUILD->STRATEGY = 'GRAND1'
617  │      append blank
618  │      replace ID with tContract+'7130'+BUILD->BLOCK
619  │      replace DS with 'Assembly of Block '+BUILD->BLOCK
620  │      replace D with BUILD->ASSY_D
621  │      replace CAL with 1
622  │   ┌──do case
623  │   ├──case BUILD->ASSY_POS = 'F'
624  │   │      replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'131'
625  │   │      replace C2 with '1131'
626  │   ├──case BUILD->ASSY_POS = 'C'
627  │   │      replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'132'
628  │   │      replace C2 with '1132'
629  │   ├──case BUILD->ASSY_POS = 'T'
630  │   │      replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'133'
631  │   │      replace C2 with '1133'
632  │   └──endcase
633  │      replace C3 with BUILD->BLK_TYPE
634  │      replace C4 with BUILD->GRAND
635  └──endif
636     * Sub-Assembly *
637     append blank
638     replace ID with tContract+'7120'+BUILD->BLOCK
639     replace DS with 'Sub-Assembly of Block '+BUILD->BLOCK
640     replace D with BUILD->ASSY_D+20
641     replace CAL with 1
642     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'12'
643     replace C2 with '112'
644     replace C3 with BUILD->BLK_TYPE
645     replace C4 with BUILD->GRAND
646     * Fabrication *
647     append blank
648     replace ID with tContract+'7110'+BUILD->BLOCK
649     replace DS with 'Fabrication of Block '+BUILD->BLOCK
650     replace D with BUILD->ASSY_D+20
651     replace CAL with 1
652     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'11'
653     replace C2 with '111'
654     replace C3 with BUILD->BLK_TYPE
655     replace C4 with BUILD->GRAND
656
657  <===return
658
659
660     ************************************************************
661     *  ACTG1 ---> Creates Activity File Data for Grand Blocks    *
662     *                                                            *
663     *                                                            *
664     ************************************************************
665
666     PROC ACTG1
667
668     * Adding ACTIVITY Records *
669     set color to w+*/r
670     @ 15, 30  SAY '--->'
671     set color to w+/r
672     select ACT
673     * Erection *
```

```
674     append blank
675     replace ID with tContract+'7160'+BUILD->BLOCK
676     replace DS with 'Erection of Grand Block '+BUILD->BLOCK
677     replace D with 20
678     replace TARGS with BUILD->ERECTION
679     replace TARGSTYPE with 'ON'
680     replace CAL with 1
681     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'141'
682     replace C2 with '1141'
683     replace C3 with BUILD->BLK_TYPE
684     replace C4 with BUILD->GRAND
685     * Post Blast O/F *
686     append blank
687     replace ID with tContract+'7254'+BUILD->BLOCK
688     replace DS with 'Post Blast O/F of GRD BLK '+BUILD->BLOCK
689     replace D with BUILD->OF4_D
690     replace CAL with 1
691     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'232'
692     replace C2 with '1232'
693     replace C3 with BUILD->BLK_TYPE
694     replace C4 with BUILD->GRAND
695
696 <===return
697
698
699     ************************************************************
700     *  ACTG2 ---> Creates Activity File Data for Grand Blocks  *
701     *                                                          *
702     *                                                          *
703     ************************************************************
704
705     PROC ACTG2
706
707     * Adding ACTIVITY Records *
708     set color to w+*/r
709     @ 15, 30  SAY '--->'
710     set color to w+/r
711     select ACT
712     * Erection *
713     append blank
714     replace ID with tContract+'7160'+BUILD->BLOCK
715     replace DS with 'Erection of Grand Block '+BUILD->BLOCK
716     replace D with 20
717     replace TARGS with BUILD->ERECTION
718     replace TARGSTYPE with 'ON'
719     replace CAL with 1
720     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'141'
721     replace C2 with '1141'
722     replace C3 with BUILD->BLK_TYPE
723     replace C4 with BUILD->GRAND
724     * Post Blast O/F *
725     append blank
726     replace ID with tContract+'7254'+BUILD->BLOCK
727     replace DS with 'Post Blast O/F of GRD BLK '+BUILD->BLOCK
728     replace D with BUILD->OF4_D
729     replace CAL with 1
730     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'232'
731     replace C2 with '1232'
732     replace C3 with BUILD->BLK_TYPE
733     replace C4 with BUILD->GRAND
734     * Blast & Painting *
735     append blank
736     replace ID with tContract+'7253'+BUILD->BLOCK
737     replace DS with 'Blast & Painting of GRD BLK '+BUILD->BLOCK
738     replace D with BUILD->OF3_D
739     replace CAL with 1
740     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'24'
741     replace C2 with '124'
742     replace C3 with BUILD->BLK_TYPE
743     replace C4 with BUILD->GRAND
744     * Pre Blast (U-R) O/F *
745     append blank
746     replace ID with tContract+'7252'+BUILD->BLOCK
747     replace DS with 'Pre Blast (U-R) O/F of GRD BLK '+BUILD->BLOCK
748     replace D with BUILD->OF2_D
```

```
749     replace CAL with 1
750     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'2312'
751     replace C2 with '12312'
752     replace C3 with BUILD->BLK_TYPE
753     replace C4 with BUILD->GRAND
754     * Pre Blast (INV) O/F *
755 ┌───if BUILD->STRATEGY = 'GRAND2' .or. BUILD->STRATEGY = 'GRAND4'
756 │       append blank
757 │       replace ID with tContract+'7251'+BUILD->BLOCK
758 │       replace DS with 'Pre Blast (INV) O/F of GRD BLK '+BUILD->BLOCK
759 │       replace D with BUILD->OF1_D
760 │       replace CAL with 1
761 │       replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'2311'
762 │       replace C2 with '12311'
763 └───endif
764     * Assembly *
765 ┌───if BUILD->STRATEGY = 'GRAND2'
766 │       append blank
767 │       replace ID with tContract+'7130'+BUILD->BLOCK
768 │       replace DS with 'Assembly of GRD BLK '+BUILD->BLOCK
769 │       replace D with BUILD->ASSY_D
770 │       replace CAL with 1
771 │       replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'131'
772 │       replace C2 with '1131'
773 │       replace C3 with BUILD->BLK_TYPE
774 │       replace C4 with BUILD->GRAND
775 └───endif
776
777 <===return
778
779
780
781
782     ****************************************************************
783     *  ACT2 ----> Creates Activity File Data for Blocks            *
784     *                                                              *
785     *                                                              *
786     ****************************************************************
787
788     PROC ACT2
789
790     * Adding ACTIVITY Records *
791     set color to w+*/r
792     @ 15, 30  SAY '--->'
793     set color to w+/r
794     select ACT
795     * Erection *
796     append blank
797     replace ID with tContract+'7150'+BUILD->BLOCK
798     replace DS with 'Stacking of Blk '+BUILD->BLOCK+' to '+BUILD->GRAND
799 ┌───do case
800 ├───case BUILD->STRATEGY = 'GRAND2'
801 │       replace D with 1
802 ├───case BUILD->STRATEGY = 'GRAND3'
803 │       replace D with 15
804 ├───case BUILD->STRATEGY = 'GRAND4'
805 │       replace D with 5
806 └───endcase
807     replace CAL with 1
808     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'142'
809     replace C2 with '1142'
810     replace C3 with BUILD->BLK_TYPE
811     replace C4 with BUILD->GRAND
812     * Assembly *
813     append blank
814     replace ID with tContract+'7130'+BUILD->BLOCK
815     replace DS with 'Assembly of Block '+BUILD->BLOCK
816     replace D with BUILD->ASSY_D
817     replace CAL with 1
818     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'131'
819     replace C2 with '1131'
820     replace C3 with BUILD->BLK_TYPE
821     replace C4 with BUILD->GRAND
822     * Sub-Assembly *
823     append blank
```

```
824     replace ID with tContract+'7120'+BUILD->BLOCK
825     replace DS with 'Sub-Assembly of Block '+BUILD->BLOCK
826     replace D with BUILD->ASSY_D+20
827     replace CAL with 1
828     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'12'
829     replace C2 with '112'
830     replace C3 with BUILD->BLK_TYPE
831     replace C4 with BUILD->GRAND
832     * Fabrication *
833     append blank
834     replace ID with tContract+'7110'+BUILD->BLOCK
835     replace DS with 'Fabrication of Block '+BUILD->BLOCK
836     replace D with BUILD->ASSY_D+20
837     replace CAL with 1
838     replace C1 with '1'+tContract+'0'+BUILD->BLOCK+'11'
839     replace C2 with '111'
840     replace C3 with BUILD->BLK_TYPE
841     replace C4 with BUILD->GRAND
842
843 <===return
844
845
846     ****************************************************************
847     *  REL1 ----> Creates Relationship Data File for Blocks        *
848     *                                                              *
849     *                                                              *
850     ****************************************************************
851
852     PROC REL1
853
854     * Adding RELATION Records *
855     set color to w+*/r
856     @ 16, 30  SAY '--->'
857     set color to w+/r
858     @ 15, 30  SAY '--->'
859     select REL
860     * Erection *
861 ┌───if BUILD->GRAND = '    '
862 │       append blank
863 │       replace ID with tContract+'7160'+BUILD->BLOCK
864 │       replace PRED with tContract+'7254'+BUILD->BLOCK
865 │       replace TYPE with 'FS'
866 │       replace LAG with 5
867 ├───else
868 │       append blank
869 │       replace ID with tContract+'7150'+BUILD->BLOCK
870 │       replace PRED with tContract+'7254'+BUILD->BLOCK
871 │       replace TYPE with 'FS'
872 │       replace LAG with 0
873 └───endif
874     * Post Blast O/F *
875     append blank
876     replace ID with tContract+'7254'+BUILD->BLOCK
877     replace PRED with tContract+'7253'+BUILD->BLOCK
878     replace TYPE with 'FS'
879     replace LAG with 0
880     * Blast & Paint *
881     append blank
882     replace ID with tContract+'7253'+BUILD->BLOCK
883     replace PRED with tContract+'7252'+BUILD->BLOCK
884     replace TYPE with 'FS'
885     replace LAG with 0
886     * Pre-Blast (U-R) O/F *
887     append blank
888     replace ID with tContract+'7252'+BUILD->BLOCK
889     replace PRED with tContract+'7251'+BUILD->BLOCK
890     replace TYPE with 'FS'
891     replace LAG with 0
892     * Pre-Blast (INV) O/F *
893     append blank
894     replace ID with tContract+'7251'+BUILD->BLOCK
895     replace PRED with tContract+'7130'+BUILD->BLOCK
896     replace TYPE with 'FS'
897     replace LAG with 5
898     * Assembly for Sub-Assembly *
```

```
899     append blank
900     replace ID with tContract+'7130'+BUILD->BLOCK
901     replace PRED with tContract+'7120'+BUILD->BLOCK
902     replace TYPE with 'SS'
903     replace LAG with 20
904     * Assembly for Fabrication *
905     append blank
906     replace ID with tContract+'7130'+BUILD->BLOCK
907     replace PRED with tContract+'7110'+BUILD->BLOCK
908     replace TYPE with 'SS'
909     replace LAG with 20
910
911
912 <===return
913
914
915     ****************************************************************
916     *  RELG1 ---> Creates Relationship Data File for               *
917     *             Grand Blocks                                     *
918     *                                                              *
919     ****************************************************************
920
921     PROC RELG1
922
923     * Adding RELATION Records *
924     set color to w+*/r
925     @ 16, 30  SAY '--->'
926     set color to w+/r
927     @ 15, 30  SAY '--->'
928     select REL
929     * Erection *
930     append blank
931     replace ID with tContract+'7160'+BUILD->BLOCK
932     replace PRED with tContract+'7254'+BUILD->BLOCK
933     replace TYPE with 'FS'
934     replace LAG with 5
935
936     * Stacking *
937     select BUILD
938     store '7254' to tStack2
939     goto tRecord
940 ┌──do while BUILD->GRAND = tGrand .and. BUILD->BLOCK < '500'
941 │   ┌──if .not. BUILD->FIRST_S = '    '
942 │   │       store BUILD->FIRST_S to tFirst_s
943 <===<===<===exit
944 │   └──endif
945 └──enddo
946     goto tRecord
947 ┌──do while BUILD->GRAND = tGrand .and. BUILD->BLOCK < '500'
948 │       select REL
949 │       append blank
950 │   ┌──if BUILD->FIRST_S = '    '
951 │   │       replace ID with tContract+'7150'+tFirst_s
952 │   ├──else
953 │   │       replace ID with tContract+tStack2+BUILD->GRAND
954 │   └──endif
955 │       replace PRED with tContract+'7150'+BUILD->BLOCK
956 │       replace TYPE with 'SS'
957 │       replace LAG with BUILD->LAG_E
958 │       select BUILD
959 │       skip
960 └──enddo
961
962
963 <===return
964
965
966
967
968     ****************************************************************
969     *  RELG2 ---> Creates Relationship Data File for               *
970     *             Grand Blocks                                     *
971     *                                                              *
972     ****************************************************************
973
```

```
 974     PROC RELG2
 975
 976     * Adding RELATION Records *
 977     set color to w+*/r
 978     @ 16, 30  SAY '--->'
 979     set color to w+/r
 980     @ 15, 30  SAY '--->'
 981     select REL
 982     * Erection *
 983     append blank
 984     replace ID with tContract+'7160'+BUILD->BLOCK
 985     replace PRED with tContract+'7254'+BUILD->BLOCK
 986     replace TYPE with 'FS'
 987     replace LAG with 5
 988     * Post Blast O/F *
 989     append blank
 990     replace ID with tContract+'7254'+BUILD->BLOCK
 991     replace PRED with tContract+'7253'+BUILD->BLOCK
 992     replace TYPE with 'FS'
 993     replace LAG with 0
 994     * Blast & Paint *
 995     append blank
 996     replace ID with tContract+'7253'+BUILD->BLOCK
 997     replace PRED with tContract+'7252'+BUILD->BLOCK
 998     replace TYPE with 'FS'
 999     replace LAG with 0
1000     * Pre-Blast (U-R) O/F *
1001 ┌──if BUILD->STRATEGY = 'GRAND2' .or. BUILD->STRATEGY = 'GRAND4'
1002 │       append blank
1003 │       replace ID with tContract+'7252'+BUILD->BLOCK
1004 │       replace PRED with tContract+'7251'+BUILD->BLOCK
1005 │       replace TYPE with 'FS'
1006 │       replace LAG with 0
1007 └──endif
1008     * Pre-Blast (INV) O/F *
1009 ┌──if BUILD->STRATEGY = 'GRAND2'
1010 │       append blank
1011 │       replace ID with tContract+'7251'+BUILD->BLOCK
1012 │       replace PRED with tContract+'7130'+BUILD->BLOCK
1013 │       replace TYPE with 'SS'
1014 │       replace LAG with 0
1015 └──endif
1016
1017     * Stacking *
1018     select BUILD
1019 ┌──do case
1020 ├──case STRATEGY = 'GRAND2'
1021 │       store '7130' to tStack2
1022 ├──case STRATEGY = 'GRAND3'
1023 │       store '7252' to tStack2
1024 ├──case STRATEGY = 'GRAND4'
1025 │       store '7251' to tStack2
1026 └──endcase
1027     goto tRecord
1028 ┌──do while BUILD->GRAND = tGrand .and. BUILD->BLOCK < '500'
1029 │   ┌──if .not. BUILD->FIRST_S = '   '
1030 │   │       store BUILD->FIRST_S to tFirst_s
1031 <===<===<===exit
1032 │   └──endif
1033 └──enddo
1034     goto tRecord
1035 ┌──do while BUILD->GRAND = tGrand .and. BUILD->BLOCK < '500'
1036 │       select REL
1037 │       append blank
1038 │   ┌──if BUILD->FIRST_S = '   '
1039 │   │       replace ID with tContract+'7150'+tFirst_s
1040 │   ├──else
1041 │   │       replace ID with tContract+tStack2+BUILD->GRAND
1042 │   └──endif
1043 │       replace PRED with tContract+'7150'+BUILD->BLOCK
1044 │   ┌──if BUILD->STRATEGY = 'GRAND3'
1045 │   │       replace TYPE with 'FS'
1046 │   ├──else
1047 │   │       replace TYPE with 'SS'
1048 │   └──endif
```

```
1049 |       replace LAG with BUILD->LAG_E
1050 |       select BUILD
1051 |       skip
1052 |   enddo
1053
1054
1055 <===return
1056
1057
1058
1059
1060     ****************************************************************
1061     *  REL2 ----> Creates Relationship Data File for Blocks        *
1062     *                                                              *
1063     *                                                              *
1064     ****************************************************************
1065
1066     PROC REL2
1067
1068     * Adding RELATION Records *
1069     set color to w+*/r
1070     @ 16, 30  SAY '--->'
1071     set color to w+/r
1072     @ 15, 30  SAY '--->'
1073     select REL
1074     * Erection *
1075     append blank
1076     replace ID with tContract+'7150'+BUILD->BLOCK
1077     replace PRED with tContract+'7130'+BUILD->BLOCK
1078     replace TYPE with 'FS'
1079     replace LAG with 0
1080     * Assembly for Sub-Assembly *
1081     append blank
1082     replace ID with tContract+'7130'+BUILD->BLOCK
1083     replace PRED with tContract+'7120'+BUILD->BLOCK
1084     replace TYPE with 'SS'
1085     replace LAG with 20
1086     * Assembly for Fabrication *
1087     append blank
1088     replace ID with tContract+'7130'+BUILD->BLOCK
1089     replace PRED with tContract+'7110'+BUILD->BLOCK
1090     replace TYPE with 'SS'
1091     replace LAG with 20
1092
1093
1094 <===return
1095
1096
1097
1098     ****************************************************************
1099     *  REL3 ----> Creates Relationship Data File for Blocks        *
1100     *                                                              *
1101     *                                                              *
1102     ****************************************************************
1103
1104     PROC REL3
1105
1106     * Adding RELATION Records *
1107     set color to w+*/r
1108     @ 16, 30  SAY '--->'
1109     set color to w+/r
1110     @ 15, 30  SAY '--->'
1111     select REL
1112     * Erection *
1113 ┌──if BUILD->GRAND = '    '
1114 |       append blank
1115 |       replace ID with tContract+'7160'+BUILD->BLOCK
1116 |       replace PRED with tContract+'7254'+BUILD->BLOCK
1117 |       replace TYPE with 'FS'
1118 |       replace LAG with 5
1119 |──else
1120 |       append blank
1121 |       replace ID with tContract+'7150'+BUILD->BLOCK
1122 |       replace PRED with tContract+'7254'+BUILD->BLOCK
1123 |       replace TYPE with 'FS'
```

```
1124 |        replace LAG with 0
1125 |    endif
1126      * Post Blast O/F *
1127      append blank
1128      replace ID with tContract+'7254'+BUILD->BLOCK
1129      replace PRED with tContract+'7253'+BUILD->BLOCK
1130      replace TYPE with 'FS'
1131      replace LAG with 0
1132      * Blast & Paint *
1133      append blank
1134      replace ID with tContract+'7253'+BUILD->BLOCK
1135      replace PRED with tContract+'7252'+BUILD->BLOCK
1136      replace TYPE with 'FS'
1137      replace LAG with 0
1138      * Pre-Blast (U-R) O/F *
1139      append blank
1140      replace ID with tContract+'7252'+BUILD->BLOCK
1141      replace PRED with tContract+'7251'+BUILD->BLOCK
1142      replace TYPE with 'FS'
1143      replace LAG with 0
1144      * Pre-Blast (INV) O/F *
1145      append blank
1146      replace ID with tContract+'7251'+BUILD->BLOCK
1147      replace PRED with tContract+'7120'+BUILD->BLOCK
1148      replace TYPE with 'FS'
1149      replace LAG with 5
1150      * Assembly for Fabrication *
1151      append blank
1152      replace ID with tContract+'7120'+BUILD->BLOCK
1153      replace PRED with tContract+'7110'+BUILD->BLOCK
1154      replace TYPE with 'SS'
1155      replace LAG with 0
1156
1157
1158 <===return
1159
1160
1161
1162
1163      ****************************************************************
1164      *  RES1 ----> Creates Resource Data File for Blocks            *
1165      *                                                              *
1166      *                                                              *
1167      ****************************************************************
1168
1169      PROC RES1
1170
1171      * Adding RESOURCE Records *
1172      set color to w+*/r
1173      @ 17, 30  SAY '--->'
1174      set color to w+/r
1175      @ 16, 30  SAY '--->'
1176      select RES
1177      * Erection *
1178 ┌──if BUILD->GRAND = '    '
1179 |        append blank
1180 |        replace ID with tContract+'7160'+BUILD->BLOCK
1181 |        replace RESCODE with 'H114'
1182 |        replace LEVEL with BUILD->BUD_ERECT
1183 |        replace LEVTYPE with 'T'
1184 |        append blank
1185 |        replace ID with tContract+'7160'+BUILD->BLOCK
1186 |        replace RESCODE with 'P_E'
1187 |        replace LEVEL with 1
1188 |        replace LEVTYPE with ' '
1189 |        replace PERIOD with 1
1190 ├──else
1191 |        append blank
1192 |        replace ID with tContract+'7150'+BUILD->BLOCK
1193 |        replace RESCODE with 'H114'
1194 |        replace LEVEL with BUILD->BUD_ERECT
1195 |        replace LEVTYPE with 'T'
1196 |        append blank
1197 |        replace ID with tContract+'7150'+BUILD->BLOCK
1198 |        replace RESCODE with 'P_E'
```

```
1199        replace LEVEL with 1
1200        replace LEVTYPE with ' '
1201        replace PERIOD with 1
1202    endif
1203    * Post Blast O/F *
1204    append blank
1205    replace ID with tContract+'7254'+BUILD->BLOCK
1206    replace RESCODE with 'H123'
1207    replace LEVEL with BUILD->BUD_OF4
1208    replace LEVTYPE with 'T'
1209    append blank
1210    replace ID with tContract+'7254'+BUILD->BLOCK
1211    replace RESCODE with 'P123'
1212    replace LEVEL with 1
1213    replace LEVTYPE with ' '
1214    * Blast & Paint *
1215    append blank
1216    replace ID with tContract+'7253'+BUILD->BLOCK
1217    replace RESCODE with 'H124'
1218    replace LEVEL with BUILD->BUD_OF3
1219    replace LEVTYPE with 'T'
1220    append blank
1221    replace ID with tContract+'7253'+BUILD->BLOCK
1222    replace RESCODE with 'P124'
1223    replace LEVEL with 1
1224    replace LEVTYPE with ' '
1225    * Pre-Blast (U-R) O/F *
1226    append blank
1227    replace ID with tContract+'7252'+BUILD->BLOCK
1228    replace RESCODE with 'H123'
1229    replace LEVEL with BUILD->BUD_OF2
1230    replace LEVTYPE with 'T'
1231    append blank
1232    replace ID with tContract+'7252'+BUILD->BLOCK
1233    replace RESCODE with 'P123'
1234    replace LEVEL with 1
1235    replace LEVTYPE with ' '
1236    * Pre-Blast (INV) O/F *
1237    append blank
1238    replace ID with tContract+'7251'+BUILD->BLOCK
1239    replace RESCODE with 'H123'
1240    replace LEVEL with BUILD->BUD_OF1
1241    replace LEVTYPE with 'T'
1242    append blank
1243    replace ID with tContract+'7251'+BUILD->BLOCK
1244    replace RESCODE with 'P123'
1245    replace LEVEL with 1
1246    replace LEVTYPE with ' '
1247    * Assembly *
1248    if BUILD->STRATEGY = 'STD' .or. BUILD->STRATEGY = 'GRAND1'
1249        append blank
1250        replace ID with tContract+'7130'+BUILD->BLOCK
1251        replace RESCODE with 'H113'
1252        replace LEVEL with BUILD->BUD_ASSY
1253        replace LEVTYPE with 'T'
1254        append blank
1255        replace ID with tContract+'7130'+BUILD->BLOCK
1256        do case
1257        case BUILD->ASSY_POS = 'F'
1258            replace RESCODE with 'P1131'
1259        case BUILD->ASSY_POS = 'C'
1260            replace RESCODE with 'P1132'
1261        case BUILD->ASSY_POS = 'T'
1262            replace RESCODE with 'P1133'
1263        endcase
1264        replace LEVEL with 1
1265        replace LEVTYPE with ' '
1266        append blank
1267        replace ID with tContract+'7130'+BUILD->BLOCK
1268        replace RESCODE with 'P_A'
1269        replace LEVEL with 1
1270        replace LEVTYPE with ' '
1271        replace PERIOD with 1
1272        replace OFFSET with BUILD->ASSY_D-1
1273    endif
```

```
1274     * Sub-Assemby *
1275     append blank
1276     replace ID with tContract+'7120'+BUILD->BLOCK
1277     replace RESCODE with 'H112'
1278     replace LEVEL with BUILD->BUD_SUB
1279     replace LEVTYPE with 'T'
1280     * Fabrication *
1281     append blank
1282     replace ID with tContract+'7110'+BUILD->BLOCK
1283     replace RESCODE with 'H111'
1284     replace LEVEL with BUILD->BUD_FAB
1285     replace LEVTYPE with 'T'
1286
1287
1288 <===return
1289
1290
1291
1292     ****************************************************************
1293     *  RESG1 ---> Creates Resource Data File for Grand Blocks      *
1294     *                                                              *
1295     *                                                              *
1296     ****************************************************************
1297
1298     PROC RESG1
1299
1300     * Adding RESOURCE Records *
1301     set color to w+*/r
1302     @ 17, 30  SAY '--->'
1303     set color to w+/r
1304     @ 16, 30  SAY '--->'
1305     select RES
1306     * Erection *
1307     append blank
1308     replace ID with tContract+'7160'+BUILD->BLOCK
1309     replace RESCODE with 'H114'
1310     replace LEVEL with BUILD->BUD_ERECT
1311     replace LEVTYPE with 'T'
1312     append blank
1313     replace ID with tContract+'7160'+BUILD->BLOCK
1314     replace RESCODE with 'P_E'
1315     replace LEVEL with 1
1316     replace LEVTYPE with ' '
1317     replace PERIOD with 1
1318     * Post Blast O/F *
1319     append blank
1320     replace ID with tContract+'7254'+BUILD->BLOCK
1321     replace RESCODE with 'H123'
1322     replace LEVEL with BUILD->BUD_OF4
1323     replace LEVTYPE with 'T'
1324     append blank
1325     replace ID with tContract+'7254'+BUILD->BLOCK
1326     replace RESCODE with 'P123'
1327     replace LEVEL with 1
1328     replace LEVTYPE with ' '
1329
1330
1331 <===return
1332
1333
1334
1335
1336
1337
1338     ****************************************************************
1339     *  RES2 ----> Creates Resource Data File for Blocks            *
1340     *                                                              *
1341     *                                                              *
1342     ****************************************************************
1343
1344     PROC RES2
1345
1346     * Adding RESOUCE Records *
1347     set color to w+*/r
1348     @ 17, 30  SAY '--->'
```

```
1349     set color to w+/r
1350     @ 16, 30  SAY '--->'
1351     select RES
1352     * Erection *
1353     append blank
1354     replace ID with tContract+'7150'+BUILD->BLOCK
1355     replace RESCODE with 'H114'
1356     replace LEVEL with BUILD->BUD_ERECT
1357     replace LEVTYPE with 'T'
1358     append blank
1359     replace ID with tContract+'7150'+BUILD->BLOCK
1360     replace RESCODE with 'P_E'
1361     replace LEVEL with 1
1362     replace LEVTYPE with ' '
1363     replace PERIOD with 1
1364     * Assembly *
1365     append blank
1366     replace ID with tContract+'7130'+BUILD->BLOCK
1367     replace RESCODE with 'H113'
1368     replace LEVEL with BUILD->BUD_ASSY
1369     replace LEVTYPE with 'T'
1370     append blank
1371     replace ID with tContract+'7130'+BUILD->BLOCK
1372     replace RESCODE with 'P1131'
1373     replace LEVEL with 1
1374     replace LEVTYPE with ' '
1375     append blank
1376     replace ID with tContract+'7130'+BUILD->BLOCK
1377     replace RESCODE with 'P_A'
1378     replace LEVEL with 1
1379     replace LEVTYPE with ' '
1380     replace PERIOD with 1
1381     replace OFFSET with BUILD->ASSY_D-1
1382     * Sub-Assemby *
1383     append blank
1384     replace ID with tContract+'7120'+BUILD->BLOCK
1385     replace RESCODE with 'H112'
1386     replace LEVEL with BUILD->BUD_SUB
1387     replace LEVTYPE with 'T'
1388     * Fabrication *
1389     append blank
1390     replace ID with tContract+'7110'+BUILD->BLOCK
1391     replace RESCODE with 'H111'
1392     replace LEVEL with BUILD->BUD_FAB
1393     replace LEVTYPE with 'T'
1394
1395
1396 <===return
1397
1398
1399
1400
1401     ****************************************************************
1402     *  RESG2 ---> Creates Resource Data Flie for Grand Blocks      *
1403     *                                                              *
1404     *                                                              *
1405     ****************************************************************
1406
1407     PROC RESG2
1408
1409     * Adding RESOURCE Records *
1410     set color to w+*/r
1411     @ 17, 30  SAY '--->'
1412     set color to w+/r
1413     @ 16, 30  SAY '--->'
1414     select RES
1415     * Erection *
1416     append blank
1417     replace ID with tContract+'7160'+BUILD->BLOCK
1418     replace RESCODE with 'H114'
1419     replace LEVEL with BUILD->BUD_ERECT
1420     replace LEVTYPE with 'T'
1421     append blank
1422     replace ID with tContract+'7160'+BUILD->BLOCK
1423     replace RESCODE with 'P_E'
```

```
1424      replace LEVEL with 1
1425      replace LEVTYPE with ' '
1426      replace PERIOD with 1
1427
1428      * Post Blast O/F *
1429      append blank
1430      replace ID with tContract+'7254'+BUILD->BLOCK
1431      replace RESCODE with 'H123'
1432      replace LEVEL with BUILD->BUD_OF4
1433      replace LEVTYPE with 'T'
1434      append blank
1435      replace ID with tContract+'7254'+BUILD->BLOCK
1436      replace RESCODE with 'P123'
1437      replace LEVEL with 1
1438      replace LEVTYPE with ' '
1439      * Blast & Paint *
1440      append blank
1441      replace ID with tContract+'7253'+BUILD->BLOCK
1442      replace RESCODE with 'H124'
1443      replace LEVEL with BUILD->BUD_OF3
1444      replace LEVTYPE with 'T'
1445      append blank
1446      replace ID with tContract+'7253'+BUILD->BLOCK
1447      replace RESCODE with 'P124'
1448      replace LEVEL with 1
1449      replace LEVTYPE with ' '
1450      * Pre-Blast (U-R) O/F *
1451      append blank
1452      replace ID with tContract+'7252'+BUILD->BLOCK
1453      replace RESCODE with 'H123'
1454      replace LEVEL with BUILD->BUD_OF2
1455      replace LEVTYPE with 'T'
1456      append blank
1457      replace ID with tContract+'7252'+BUILD->BLOCK
1458      replace RESCODE with 'P123'
1459      replace LEVEL with 1
1460      replace LEVTYPE with ' '
1461      * Pre-Blast (INV) O/F *
1462   ┌──if BUILD->STRATEGY = 'GRAND2' .or. BUILD->STRATEGY = 'GRAND4'
1463   │      append blank
1464   │      replace ID with tContract+'7251'+BUILD->BLOCK
1465   │      replace RESCODE with 'H123'
1466   │      replace LEVEL with BUILD->BUD_OF1
1467   │      replace LEVTYPE with 'T'
1468   │      append blank
1469   │      replace ID with tContract+'7251'+BUILD->BLOCK
1470   │      replace RESCODE with 'P123'
1471   │      replace LEVEL with 1
1472   │      replace LEVTYPE with ' '
1473   └──endif
1474      * Assembly *
1475   ┌──if BUILD->STRATEGY = 'GRAND2'
1476   │      append blank
1477   │      replace ID with tContract+'7130'+BUILD->BLOCK
1478   │      replace RESCODE with 'H113'
1479   │      replace LEVEL with BUILD->BUD_ASSY
1480   │      replace LEVTYPE with 'T'
1481   │      append blank
1482   │      replace ID with tContract+'7130'+BUILD->BLOCK
1483   │   ┌──do case
1484   │   ├──case BUILD->ASSY_POS = 'F'
1485   │   │      replace RESCODE with 'P1131'
1486   │   ├──case BUILD->ASSY_POS = 'C'
1487   │   │      replace RESCODE with 'P1132'
1488   │   ├──case BUILD->ASSY_POS = 'T'
1489   │   │      replace RESCODE with 'P1133'
1490   │   └──endcase
1491   │      replace LEVEL with 1
1492   │      replace LEVTYPE with ' '
1493   │      append blank
1494   │      replace ID with tContract+'7130'+BUILD->BLOCK
1495   │      replace RESCODE with 'P_A'
1496   │      replace LEVEL with 1
1497   │      replace LEVTYPE with ' '
1498   │      replace PERIOD with 1
```

```
1499 |       replace OFFSET with BUILD->ASSY_D-1
1500 |    endif
1501
1502
1503 <===return
1504
1505
1506
1507
1508     ****************************************************************
1509     *  MODEL_M -> Text for Block Build Menu                        *
1510     *                                                              *
1511     *                                                              *
1512     ****************************************************************
1513
1514     PROC MODEL_M
1515
1516     set color to gr+/b
1517     @  4, 17  to  18, 58 double
1518     @  4, 30  SAY " MODEL BUILD MENU "
1519     set color to w+/r
1520     @  5, 18  SAY '                                          '
1521     @  6, 18  SAY '                                          '
1522     @  7, 18  SAY '     Enter Contract Letter : :            '
1523     @  8, 18  SAY '     Enter Project Name w/Path            '
1524     @  9, 18  SAY '                                          '
1525     @ 10, 18  SAY '                                          '
1526     @ 11, 18  SAY '     Enter Build Data Base w/Path         '
1527     @ 12, 18  SAY '                                          '
1528     @ 13, 18  SAY '                                          '
1529     @ 14, 18  SAY '                                          '
1530     @ 15, 18  SAY '                                          '
1531     @ 16, 18  SAY '                                          '
1532     @ 17, 18  SAY '                                          '
1533     set color to gr
1534     @ 19, 26  say 'Press "Esc" to Return'
1535
1536 <===return
1537
1538
1539
1540     *Formatted by: dANALYST Ver. 7.3a on March 24, 1992 at 9:47 AM.
dANALYST found 0 error(s), 0 warning(s), 1540 lines.
```

```
INPUT FILE:  C:\OPLAN\SP8_DATA.PRG
    1     *  FILE NAME: SP8_DATA.PRG
    2     *  BY: D. McQuaide
    3     *  DATE: March 24, 1992
    4     *  DESC:
    5     *  CALLED BY:
    6     *  DATA FILES:
    7     * SP8_DATA.prg
    8     *
    9
   10     set procedure to SP8_DATA
   11
   12     ********************************************************************************
   13     store space(30) to tAgg
   14     store space(30) to tCurve
   15     set color to gr+/b
   16     @  4, 17  to  18, 58 double
   17     @  4, 30  SAY " CURVE GENERATION "
   18     set color to w+/r
   19     @  5, 18  SAY '                                          '
   20     @  6, 18  SAY '                                          '
   21     @  7, 18  SAY '     Enter Aggregation File w/Path        '
   22     @  8, 18  SAY '                                          '
   23     @  9, 18  SAY '                                          '
   24     @ 10, 18  SAY '     Enter Curve File w/Path              '
   25     @ 11, 18  SAY '                                          '
   26     @ 12, 18  SAY '                                          '
   27     @ 13, 18  SAY '                                          '
   28     @ 14, 18  SAY '                                          '
   29     @ 15, 18  SAY '                                          '
   30     @ 16, 18  SAY '                                          '
   31     @ 17, 18  SAY '                                          '
   32     set color to gr
   33     @ 19, 26  say 'Press "Esc" to Return'
   34     ********************************************************************************
   35  ┌──do while .T.
   36  │     @  8, 23  get tAgg picture '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
   37  │     read
   38  │  ┌──if readkey() = 12
   39  <===<===<===return
   40  │  └──endif
   41  │     store trim(tAgg)+'.agg' to dAgg
   42  │  ┌──if .not. file(dAgg)
   43  │  │     set color to gr+/r
   44  │  │     @  9, 18  SAY '          File NOT Found              '
   45  <===<===<===loop
   46  │  ├──else
   47  <===<===<===exit
   48  │  └──endif
   49  └──enddo
   50     set color to gr+/r
   51     @  9, 18  SAY '                                          '
   52
   53     ********************************************************************************
   54  ┌──do while .T.
   55  │     @ 11, 23  get tCurve picture '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
   56  │     read
   57  │  ┌──if readkey() = 12
   58  <===<===<===return
   59  │  └──endif
   60  │     store trim(tCurve)+'.dbf' to dCurve
   61  │  ┌──if .not. file(dCurve)
   62  │  │     set color to gr+/r
   63  │  │     @ 12, 18  SAY '          File NOT Found              '
   64  <===<===<===loop
   65  │  ├──else
   66  <===<===<===exit
   67  │  └──endif
   68  └──enddo
   69     ********************************************************************************
   70     set color to gr+/r
   71     @ 12, 18  SAY '                                          '
   72
   73     select B
```

text display for file names

read name of AGGREGATION and verifies file exists

read name of CURVE File and verifies file exists

Set-up files

```
 74     use &dAgg index TODATE alias AGG
 75     reindex
 76
 77     select A
 78     use &dCurve alias CURVE
 79
 80     ********************************************************************************
 81     store ctod(' ') to tDate
 82     store 0 to tPeriod
 83     store 0 to tDays
 84     @ 13, 18  SAY '    Enter # of Day /Period :           '
 85     @ 14, 18  SAY '          Enter Start Date :           '
 86     @ 15, 18  SAY '   Enter Number of Periods :           '
 87  ┌──do while .T.
 88  │      @ 13, 47  get tDays picture '99'
 89  │      @ 14, 47  get tDate
 90  │      @ 15, 47  get tPeriod picture '999'
 91  │      read
 92  │   ┌──if readkey() = 12
 93  <===<===<===return
 94  │   └──endif
 95  │      store tDate + 6 to tStart
 96  │      store (tDate + (tDays * tPeriod)) - 1 to tFinish
 97  │      select B
 98  │      seek tStart
 99  │   ┌──if eof()
100  │   │      set color to w+/r
101  │   │      @ 16, 18  SAY '       Starting Date                  '
102  │   │      set color to w+*/r
103  │   │      @ 16, 40  SAY 'Not Found'
104  │   │      set color to w+/r
105  <===<===<===loop
106  │   └──endif
107  │      set color to w+/r
108  │      @ 16, 18  SAY '                                          '
109  <===<===exit
110  └──enddo
111
112    ********************************************************************************
113    select AGG
114    copy to SP8_TEMP fields TODATE,WORKPDS for recno() < tPeriod+1
115    set index to RES_DATE
116    reindex
117
118    select CURVE
119    zap
120    append from SP8_TEMP
121    set index to TODATE
122    reindex
123
124    select AGG
125    set relation to TODATE into CURVE
126    set color to gr+/r
127
128    ********************************************************************************
129    ***** Creating Curve Data *****
130
131
132    @ 17, 18  SAY '    Creating Data for Rescode H111      '
133    store 'H111' to tRescode
134    store 'CURVE->H111' to tField
135    do DATA1
136
137    @ 17, 18  SAY '    Creating Data for Rescode H112      '
138    store 'H112' to tRescode
139    store 'CURVE->H112' to tField
140    do DATA1
141
142    @ 17, 18  SAY '    Creating Data for Rescode H1131     '
143    store 'H1131' to tRescode
144    store 'CURVE->H1131' to tField
145    do DATA1
146
147    @ 17, 18  SAY '    Creating Data for Rescode H1132     '
148    store 'H1132' to tRescode
```

read program parameters
- days in period
- starting date
- number of periods

builds new data file based on above paramaters

```
149     store 'CURVE->H1132' to tField
150     do DATA1
151
152     @ 17, 18  SAY '    Creating Data for Rescode H1133     '
153     store 'H1133' to tRescode
154     store 'CURVE->H1133' to tField
155     do DATA1
156
157     @ 17, 18  SAY '    Creating Data for Rescode H1141     '
158     store 'H1141' to tRescode
159     store 'CURVE->H1141' to tField
160     do DATA1
161
162     @ 17, 18  SAY '    Creating Data for Rescode H1142     '
163     store 'H1142' to tRescode
164     store 'CURVE->H1142' to tField
165     do DATA1
166
167     @ 17, 18  SAY '    Creating Data for Rescode H12311    '
168     store 'H12311' to tRescode
169     store 'CURVE->H12311' to tField
170     do DATA1
171
172     @ 17, 18  SAY '    Creating Data for Rescode H12312    '
173     store 'H12312' to tRescode
174     store 'CURVE->H12312' to tField
175     do DATA1
176
177     @ 17, 18  SAY '    Creating Data for Rescode H1232     '
178     store 'H1232' to tRescode
179     store 'CURVE->H1232' to tField
180     do DATA1
181
182     @ 17, 18  SAY '    Creating Data for Rescode H124      '
183     store 'H124' to tRescode
184     store 'CURVE->H124' to tField
185     do DATA1
186
187     @ 17, 18  SAY '    Creating Data for Rescode P1131     '
188     store 'P1131' to tRescode
189     store 'CURVE->P1131' to tField
190     do DATA2
191
192     @ 17, 18  SAY '    Creating Data for Rescode P1132     '
193     store 'P1132' to tRescode
194     store 'CURVE->P1132' to tField
195     do DATA2
196
197     @ 17, 18  SAY '    Creating Data for Rescode P1133     '
198     store 'P1133' to tRescode
199     store 'CURVE->P1133' to tField
200     do DATA2
201
202     @ 17, 18  SAY '    Creating Data for Rescode P12311    '
203     store 'P12311' to tRescode
204     store 'CURVE->P12311' to tField
205     do DATA2
206
207     @ 17, 18  SAY '    Creating Data for Rescode P12312    '
208     store 'P12312' to tRescode
209     store 'CURVE->P12312' to tField
210     do DATA2
211
212     @ 17, 18  SAY '    Creating Data for Rescode P1232     '
213     store 'P1232' to tRescode
214     store 'CURVE->P1232' to tField
215     do DATA2
216
217     @ 17, 18  SAY '    Creating Data for Rescode P124      '
218     store 'P124' to tRescode
219     store 'CURVE->P124' to tField
220     do DATA2
221
222     @ 17, 18  SAY '    Creating Data for Rescode P_A       '
223     store 'P_A' to tRescode
```

store resource codes and field name to variables used in Data 1 and Data 2 sub routi

```
224     store 'CURVE->P_A' to tField
225     do DATA2
226
227     @ 17, 18  SAY '    Creating Data for Rescode P_E       '
228     store 'P_E' to tRescode
229     store 'CURVE->P_E' to tField
230     do DATA2
231
232     ***************************************************************************
233     select CURVE
234
235     @ 17, 18  SAY '    Creating Data for Rescode H113      '
236     replace all H113 with H1131+H1132+H1133
237
238     @ 17, 18  SAY '    Creating Data for Rescode H114      '
239     replace all H114 with H1141+H1142
240
241     @ 17, 18  SAY '    Creating Data for Rescode H1231     '
242     replace all H1231 with H12311+H12312
243
244     @ 17, 18  SAY '    Creating Data for Rescode H123      '
245     replace all H123  with H1231 +H1232
246
247     @ 17, 18  SAY '    Creating Data for Rescode H11       '
248     replace all H11   with H111+H112+H113+H114
249
250     @ 17, 18  SAY '    Creating Data for Rescode H12       '
251     replace all H12   with H123+H124
252
253     @ 17, 18  SAY '    Creating Data for Rescode H1        '
254     replace all H1    with H11+H12
255
256     @ 17, 18  SAY '    Creating Data for Rescode P113      '
257     replace all P113  with P1131+P1132+P1133
258
259     @ 17, 18  SAY '    Creating Data for Rescode P1231     '
260     replace all P1231 with P12311+P12312
261
262     @ 17, 18  SAY '    Creating Data for Rescode P123      '
263     replace all P123  with P1231+P1232
264
265     @ 17, 18  SAY '    Creating Data for Rescode P12       '
266     replace all P12   with P123+P124
267     ***************************************************************************
268
269 <===return
270
271
272     ************************************
273     *          PROCEDURE DATA1         *
274     *  Calculates Manning Requirements  *
275     ************************************
276
277     PROC DATA1
278
279     store 0 to temp_Man
280     seek tRescode
281 ┌──if .not. eof()
282 │     ┌──do while RESCODE = tRescode .and. .not. eof()
283 │     │     ┌──if WORKPDS = 0
284 │     │     │       replace &tField with temp_Man
285 │     │     ├──else
286 │     │     │       replace &tField with (LREQTOT / 8) / WORKPDS
287 │     │     │       store (LREQTOT / 8) / WORKPDS to temp_Man
288 │     │     └──endif
289 │     │        skip
290 │     └──enddo
291 └──endif
292 <===return
293
294
295
296     ************************************
297     *          PROCEDURE DATA2         *
298     *  Calculates Laydown Requirements  *
```

calculates summary Manning

```
299     ************************************
300
301     PROC DATA2
302
303     store 0 to temp_Man
304     seek tRescode
305 ┌───if .not. eof()
306 │     ┌──do while RESCODE = tRescode .and. .not. eof()
307 │     │     ┌──if WORKPDS = 0
308 │     │     │        replace &tField with temp_Man
309 │     │     ├──else
310 │     │     │        replace &tField with (LREQTOT) / WORKPDS
311 │     │     │        store (LREQTOT) / WORKPDS to temp_Man
312 │     │     └──endif
313 │     │        skip
314 │     └──enddo
315 └───endif
316 <===return
317
318
319
320
321
322
323
324     *Formatted by: dANALYST Ver. 7.3a on March 24, 1992 at 9:47 AM.
dANALYST found 0 error(s), 0 warning(s), 324 lines.
```

```
INPUT FILE:  C:\OPLAN\SP8_GNT.PRG
    1     *  FILE NAME: SP8_GNT.PRG
    2     *  BY: D. McQuaide
    3     *  DATE: March 24, 1992
    4     *  DESC:
    5     *  CALLED BY:
    6     *  DATA FILES:
    7     * SP8_GNT.prg
    8
    9     set procedure to SP8_GNT
   10
   11     ********************************************************************************
   12     * Setting Up Print Variables ***
   13     store chr(027)+'E' to pReset
   14     store chr(027)+'&l0O' to pPort
   15     store chr(027)+'&l6D' to p6lpi
   16     store chr(027)+'&l8D' to p8lpi
   17     store chr(027)+'&l2A' to pLetter
   18     store chr(027)+'(8Q'+chr(027)+'(s0p16.67h8.4v0s0b3T' to pFont
   19     store chr(027)+'(0U'+chr(027)+'(s0p10.00h13.9v0s3b11T' to pFont1
   20     store chr(027)+'(1U'+chr(027)+'(s0p6.53h18.0v0s3b11T' to pFont2
   21     store chr(027)+'(8U'+chr(027)+'(s0p12.00h12.0v0s0b6T' to pFont3
   22     store chr(027)+'(10U'+chr(027)+'(s0p10.00h12.0v0s0b3T' to pFont4
   23     store chr(027)+'&d3D' to U_on
   24     store chr(027)+'&d@' to U_off
   25     ********************************************************************************
   26
   27     do MENU_1
   28     store space(30) to tAct
   29  ┌──do while .T.
   30  │     set color to w+/r
   31  │     @  6, 18  SAY '     Enter Model Name w/Path            '
   32  │     @  7, 18  SAY '                                        '
   33  │     @  7, 23  get tAct picture '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
   34  │     read
   35  │  ┌──if readkey() = 12
   36  <===<===<===return
   37  │  └──endif
   38  │     store trim(tAct)+'.act' to tActivity
   39  │     store trim(tAct)+'.rel' to tResource
   40  │  ┌──if file(tActivity)
   41  │  │     set color to w/r
   42  │  │     @  6, 18  SAY '                                        '
   43  │  │     @  7, 18  SAY '                                        '
   44  │  │     @  8, 18  SAY '                                        '
   45  │  │     @  9, 18  SAY '                                        '
   46  <===<===<===exit
   47  │  ├──else
   48  │  │     set color to w+/r
   49  │  │     @  9, 18  SAY '          Can NOT Find File             '
   50  <===<===<===loop
   51  │  └──endif
   52  └──enddo
   53     ********************************************************************************
   54
   55     * Setting up Data Bases *
   56     set color to w+/r
   57     @ 16, 18  SAY '       Setting Up Data Bases          '
   58     store 'C:\DATA\DBASE\BASE\NASSCO_D.dbf' to tCalendar
   59     store 'C:\DATA\DBASE\BASE\DATES.ndx'    to tNDX_D1
   60     store 'C:\DATA\DBASE\BASE\NASSCO.ndx'   to tNDX_D2
   61     store 'C:\DATA\DBASE\SP8\T_RECORD.dbf'  to tT_Record
   62     store 'C:\DATA\DBASE\SP8\T_HOURS.dbf'   to tT_Hours
   63
   64     select C
   65     use &tResource alias Resource
   66     set index to ID
   67     set color to w*/r
   68     @ 17, 31  SAY 'Re-Indexing'
   69     reindex
   70     set color to w/r
   71     @ 17, 31  SAY '           '
   72
   73     select B
```

set up print variables

read project name and verify file exist

set up data files

```
74      use &tCalendar alias Calendar
75
76      select A
77      use &tActivity alias Activity
78      set index to GANTT
79      set color to w*/r
80      @ 17, 31  SAY 'Re-Indexing'
81      reindex
82      set color to w/r
83      @ 16, 18  SAY '                                          '
84      @ 17, 31  SAY '           '
85
86      ********************************************************************************
87      * Starting the Gantt Program *
88  ┌──do while .T.
89  │       store ' '   to tContract
90  │       store '   ' to tBlock
91  │       set color to gr
92  │       clear
93  │       do MENU_2
94  │   ┌──do while .T.
95  │   │       @  7, 46 get tContract picture '!'
96  │   │       @  8, 40 get tBlock    picture '!!!'
97  │   │       read
98  │   │   ┌──if readkey() = 12
99  <===<===<===<===return
100 │   │   └──endif
101 │   │       select Activity
102 │   │       set color to w+/r
103 │   │       @ 16, 18  SAY '                                          '
104 │   │       seek tContract+tBlock
105 │   │   ┌──if eof()
106 │   │   │       set color to w+/r
107 │   │   │       @ 16, 18  SAY '       Record           in Model        '
108 │   │   │       set color to w+*/r
109 │   │   │       @ 16, 32  SAY 'Record Not'
110 <===<===<===<===loop
111 │   │   └──endif
112 │   │       store LSDATE - 20 to WK_Start
113 <===<===<===exit
114 │   └──enddo
115 │       ********************************************************************************
116 │       select Activity
117 │       * Setting Up Memory Variables *
118 │       * Date Bar Headers *
119 │       store ' ' to tHEAD1
120 │       store ' ' to tHEAD2
121 │       store ' ' to tHEAD3
122 │       store ctod(' ') to S_Date
123 │       * Weekly Background Variables *
124 │       store chr(176)+chr(176)+chr(176) to s3
125 │       store chr(176)+chr(176)+chr(176)+chr(176) to s4
126 │       store chr(176)+chr(176)+chr(176)+chr(176)+chr(176) to s5
127 │       store '   ' to b3
128 │       store '    ' to b4
129 │       store '     ' to b5
130 │       ********************************************************************************
131 │
132 │       store 'M/V WELL PLANNED' to Hullname
133 │       do DATE_BAR
134 │       do GANTT_1
135 │       select D
136 │       use
137 │
138 │       ********************************************************************************
139 └──enddo
140
141 <===return
142
143
144     ***************************
145     * PROCEDURE GANTT_1       *
146     ***************************
147
148     PROC GANTT_1
```

read contract letter and bl
number, verify record exis

set up variables us
in sub-routines

call out sub-routines

```
149
150     ********************************************************************************
151     * Setting up Memory Variables *
152     * Steel *
153     store ctod(' ') to t711s,t711f
154     store ctod(' ') to t712s,t712f
155     store ctod(' ') to t713s,t713f
156     store ctod(' ') to t715s,t715f
157     store ctod(' ') to t716s,t716f
158     store 0 to t711d,t712d,t713d,t715d,t716d
159     store ' ' to tStack
160     * Outfitting *
161     store ctod(' ') to t7251s,t7251f
162     store ctod(' ') to t7252s,t7252f
163     store ctod(' ') to t7253s,t7253f
164     store ctod(' ') to t7254s,t7254f
165     store 0 to t7251d,t7252d,t7253d,t7254d
166     ********************************************************************************
167   do while substr(ID,1,1) = tContract .and. substr(ID,6,3) = tBlock
168     do case
169     case substr(ID,2,3) = '711'
170          store D to t711d
171          store LSDATE to t711s
172          store LFDATE to t711f
173     case substr(ID,2,3) = '712'
174          store D to t712d
175          store LSDATE to t712s
176          store LFDATE to t712f
177     case substr(ID,2,3) = '713'
178          store D to t713d
179          store LSDATE to t713s
180          store LFDATE to t713f
181     case substr(ID,2,3) = '715'
182          store D to t715d
183          store LSDATE to t715s
184          store LFDATE to t715f
185          store 'T' to tStack
186     case substr(ID,2,3) = '716'
187          store D to t716d
188          store LSDATE to t716s
189          store LFDATE to t716f
190     case substr(ID,2,4) = '7251'
191          store D to t7251d
192          store LSDATE to t7251s
193          store LFDATE to t7251f
194     case substr(ID,2,4) = '7252'
195          store D to t7252d
196          store LSDATE to t7252s
197          store LFDATE to t7252f
198     case substr(ID,2,4) = '7253'
199          store D to t7253d
200          store LSDATE to t7253s
201          store LFDATE to t7253f
202     case substr(ID,2,4) = '7254'
203          store D to t7254d
204          store LSDATE to t7254s
205          store LFDATE to t7254f
206     endcase
207       skip
208   enddo
209     ********************************************************************************
210
211     * Calculating Floats
212     select Calendar
213     set index to &tNDX_D1
214     seek t713f
215     store NASSCOD to Temp_D1
216     seek t7251s
217     store NASSCOD to Temp_D2
218     * Assembly Float *
219     store (Temp_D2 - Temp_D1) - 1 to Assy_Float
220     seek t7254f
221     store NASSCOD to Temp_D3
222   if tStack = ' '
223        seek t716s
```

set up start/complete variables

load dates to variables

```
224         store NASSCOD to Temp_D4
225     else
226         seek t715s
227         store NASSCOD to Temp_D4
228     endif
229     * Assembly Float *
230     store (Temp_D4 - Temp_D3) - 1 to OF_Float
231     ************************************************************************************
232
233     do SCREEN1          &&---------->>>>>>> display schedule on screen
234
235     ************************************************************************************
236     store 'N' to tPrint
237     set color to gr+/b
238     @ 22,19 say 'Print Hard Copy of GANTT Chart (Y/N) : :'
239     @ 22,57 get tPrint picture '!'
240     read
241     if tPrint = 'Y'
242         set color to w+*
243         @ 23, 33 say 'PRINTING'
244         if tBlock > '500'
245             select ACTIVITY
246             copy to TEMP_ACT for ID = tContract .and. C4 = tBlock
247             select D
248             use TEMP_ACT alias TEMP_ACT
249             set index to C_BLK_S
250             reindex
251             go top
252             set device to print
253             @   1,   1 say pReset
254             @   1,   1 say pPort
255             @   1,   1 say pOlpi
256             @   1,   1 say pLetter
257             do G_PAGE_D
258             do G_PAGE_G
259             set device to screen
260             use
261         else
262             select ACTIVITY
263             copy to TEMP_ACT for ID = tContract .and. substr(ID,6,3) = tBlock
264             select D
265             use TEMP_ACT alias TEMP_ACT
266             set index to C_BLK_S
267             reindex
268             go top
269             set device to print
270             @   1,   1 say pReset
271             @   1,   1 say pPort
272             @   1,   1 say pOlpi
273             @   1,   1 say pLetter
274             do G_PAGE_A
275             set device to screen
276         endif
277         set color to gr
278         @ 23, 31 say '        '
279     endif
280     ************************************************************************************
281
282 <===return
283
284
285     *************************
286     * PROCEDURE G_PAGE_A    *
287     *************************
288
289     PROC G_PAGE_A
290
291     ************************************************************************************
292     store 8 to x
293     store 1 to y
294     @ x,y+10  say pFont2+'Block '+tBlock+' Build Strategy'+pFont4
295     @ x,y+10  say ' '
296     x = x + 2
297     @  x,  2 say U_on+'STEEL'
298     @  x, 17 say 'Start'
```

print schedule

```
299     @  x, 26 say 'Comp'
300     @  x, 35 say 'Dur'+U_off
301     @  x, 35 say ' '
302     @  x, 40 say U_on+'OUTFITTING'
303     @  x, 56 say 'Start'
304     @  x, 65 say 'Comp'
305     @  x, 74 say 'Dur'+U_off
306     @  x, 74 say ' '
307     x = x + 1
308     ********************************************************************************
309     @  x,  8 say 'Fab(711)'
310  ---if t711d > 0
311  |      @  x, 17 say t711s
312  |      @  x, 26 say t711f
313  |      @  x, 35 say t711d picture '999'
314  ---endif
315     @  x, 41 say 'O/F Inv.(7251)'
316  ---if t7251d > 0
317  |      @  x, 56 say t7251s
318  |      @  x, 65 say t7251f
319  |      @  x, 74 say t7251d picture '999'
320  ---endif
321     x = x + 1
322     @  x,  3 say 'Sub-Assy(712)'
323  ---if t712d > 0
324  |      @  x, 17 say t712s
325  |      @  x, 26 say t712f
326  |      @  x, 35 say t712d picture '999'
327  ---endif
328     @  x, 41 say 'O/F Up-R(7252)'
329  ---if t7252d > 0
330  |      @  x, 56 say t7252s
331  |      @  x, 65 say t7252f
332  |      @  x, 74 say t7252d picture '999'
333  ---endif
334     x = x + 1
335     @  x,  3 say 'Assembly(713)'
336  ---if t713d > 0
337  |      @  x, 17 say t713s
338  |      @  x, 26 say t713f
339  |      @  x, 35 say t713d picture '999'
340  ---endif
341     @  x, 42 say 'S/B & P(7253)'
342  ---if t7253d > 0
343  |      @  x, 56 say t7253s
344  |      @  x, 65 say t7253f
345  |      @  x, 74 say t7253d picture '999'
346  ---endif
347     x = x + 1
348  ---if t715d = 0
349  |      @  x,  3 say 'Erection(716)'
350  |   ---if t716d > 0
351  |   |      @  x, 17 say t716s
352  |   |      @  x, 26 say t716f
353  |   |      @  x, 35 say t716d picture '999'
354  |   ---endif
355  ---else
356  |      @  x,  3 say 'Stacking(715)'
357  |   ---if t715d > 0
358  |   |      @  x, 17 say t715s
359  |   |      @  x, 26 say t715f
360  |   |      @  x, 35 say t715d picture '999'
361  |   ---endif
362  ---endif
363     @  x, 41 say 'O/F Post(7254)'
364  ---if t7254d > 0
365  |      @  x, 56 say t7254s
366  |      @  x, 65 say t7254f
367  |      @  x, 74 say t7254d picture '999'
368  ---endif
369     x = x + 1
370     @  x, 20 say 'Assembly Float'
371     @  x, 35 say ASSY_FLOAT picture '999'
372     @  x, 57 say 'Outfitting Float'
373     @  x, 74 say OF_FLOAT picture '999'
```

prints headings for date portion of Gantt Schedules

print schedule dates for date portion of the Gantt Schedule

```
374     x = x + 4
375     *************************************************************************
376
377     go top
378     store ' ' to tHEAD1
379     store ' ' to tHEAD2
380     store ' ' to tHEAD3
381     store ' ' to tSHADE1
382     store ctod(' ') to S_date
383     store LSDATE-20 to WK_Start
384     do DATE_BAR
385     store 45 to y
386     @ x,y     say pFont
387     @ x,y     say p8lpi
388     @ x,y-32  say chr(201)
389     @ x,y-31  say replicate(chr(205),100)
390     @ x,y+69  say chr(187)
391     x = x + 1
392     @ x,y-32  say chr(186)
393     @ x,y     say tHEAD1
394     @ x,y+69  say chr(186)
395     x = x + 1
396     @ x,y-32  say chr(186)
397     @ x,y     say tHEAD2
398     @ x,y+69  say chr(186)
399     x = x + 1
400     @ x,y-32  say chr(186)
401     @ x,y     say tHEAD3
402     @ x,y+69  say chr(186)
403     x = x + 1
404     *************************************************************************
405     do while .not. eof()
406       if D > 0
407           @ x,y-32  say chr(186)
408           @ x,y     say tSHADE1
409           @ x,y+69  say chr(186)
410           x = x + 1
411           store ' ' to tErect
412           @ x,y     say tSHADE1
413         do case
414         case substr(ID,2,4) = '7110'
415             store 'Steel         Fabrication' to tDescrpt
416             store chr(220) to tGantt
417         case substr(ID,2,4) = '7120'
418             store 'Steel        Sub-Assembly' to tDescrpt
419             store chr(220) to tGantt
420         case substr(ID,2,4) = '7130'
421             store 'Steel            Assembly' to tDescrpt
422             store chr(220) to tGantt
423         case substr(ID,2,4) = '7251'
424             store 'O/F  Pre-Blast (Inverted)' to tDescrpt
425             store chr(207) to tGantt
426         case substr(ID,2,4) = '7252'
427             store 'O/F  Pre-Blast (Up-Right)' to tDescrpt
428             store chr(209) to tGantt
429         case substr(ID,2,4) = '7253'
430             store 'O/F         Blast & Paint' to tDescrpt
431             store chr(127) to tGantt
432         case substr(ID,2,4) = '7254'
433             store 'O/F            Post Blast' to tDescrpt
434             store chr(223) to tGantt
435         case substr(ID,2,4) = '7150'
436             store 'Steel            Stacking' to tDescrpt
437             store chr(177) to tGantt
438             store chr(004) to tE_Date
439             store 'Y'      to tErect
440         case substr(ID,2,4) = '7160'
441             store 'Steel           Erection' to tDescrpt
442             store chr(177) to tGantt
443             store chr(004) to tE_Date
444             store 'Y'      to tErect
445         endcase
446         store (LFDATE-LSDATE)+1 to tempDur
447         store (LSDATE-S_date)+1 to tempDur1
448         tGanttD = iif(mod(tempDur,7) > 0,int(tempDur/7) + 1,int(tempDur/7))
```

prints date bar heading Gantt Schedule

prints Gantt bars for Gantt Schedule

```
449             tStartD = iif(mod(tempDur1,7) > 0,int(tempDur1/7) + 1,int(tempDur1/7))
450          if tErect = ' '
451             @ x,y-32  say chr(186)
452             @ x,y-30  say tDescrpt
453             @ x,y+tStartD say replicate (tGantt,tGanttD)
454             @ x,y+tStartD-10 say LSDATE
455             @ x,y+tStartD+(tGanttD+2) say LFDATE
456             @ x,y+69  say chr(186)
457             x = x + 1
458          else
459             @ x,y-32  say chr(186)
460             @ x,y-30  say tDescrpt
461             @ x,y+tStartD say '*'
462             @ x,y+tStartD say replicate (tGantt,tGanttD)
463             @ x,y+tStartD-10 say LSDATE
464             @ x,y+tStartD+(tGanttD+2) say LFDATE
465             @ x,y+69  say chr(186)
466             @ x,y+(((LSDATE-S_date)/7)+1) say '*'
467             x = x + 1
468          endif
469       endif
470       skip
471    enddo
472    @ x,y-32  say chr(200)
473    @ x,y-31  say replicate(chr(205),100)
474    @ x,y+69  say chr(188)
475    ********************************************************************************
476
477 <===return
478
479
480    ***************************
481    *   PROCEDURE G_PAGE_D    *
482    *    for Grand Blocks     *
483    ***************************
484
485    PROC G_PAGE_D
486
487    ********************************************************************************
488    store 1 to x
489    store 1 to y
490    @ x,y+10  say pFont2+'Grand Block '+C4+' Build Strategy'+pFont4
491    @ x,y+10  say ' '
492    x = x + 2
493    do while .not. eof()
494       store substr(ID,6,3) to tempBlk
495       do G_BLK1
496    enddo
497    ********************************************************************************
498
499 <===return
500
501
502    ***************************
503    *    PROCEDURE G_BLK1     *
504    *    for Grand Blocks     *
505    ***************************
506
507    PROC G_BLK1
508
509    ********************************************************************************
510    * Setting up Memory Variables *
511    * Steel *
512    store ctod(' ') to t711s,t711f
513    store ctod(' ') to t712s,t712f
514    store ctod(' ') to t713s,t713f
515    store ctod(' ') to t715s,t715f
516    store ctod(' ') to t716s,t716f
517    store 0 to t711d,t712d,t713d,t715d,t716d
518    store ' ' to tStack
519    * Outfitting *
520    store ctod(' ') to t7251s,t7251f
521    store ctod(' ') to t7252s,t7252f
522    store ctod(' ') to t7253s,t7253f
523    store ctod(' ') to t7254s,t7254f
```

set up to print grand block Gantt Schedule

sets up start/complete variables

```
524     store 0 to t7251d,t7252d,t7253d,t7254d
525     *****************************************************************************
526  ┌──do while substr(ID,6,3) = temp8lk .and. .not. eof()
527  │    ┌──do case
528  │    ├──case substr(ID,2,3) = '711'
529  │    │       store D to t711d
530  │    │       store LSDATE to t711s
531  │    │       store LFDATE to t711f
532  │    ├──case substr(ID,2,3) = '712'
533  │    │       store D to t712d
534  │    │       store LSDATE to t712s
535  │    │       store LFDATE to t712f
536  │    ├──case substr(ID,2,3) = '713'
537  │    │       store D to t713d
538  │    │       store LSDATE to t713s
539  │    │       store LFDATE to t713f
540  │    ├──case substr(ID,2,3) = '715'
541  │    │       store D to t715d
542  │    │       store LSDATE to t715s
543  │    │       store LFDATE to t715f
544  │    │       store 'T' to tStack
545  │    ├──case substr(ID,2,3) = '716'
546  │    │       store D to t716d
547  │    │       store LSDATE to t716s
548  │    │       store LFDATE to t716f
549  │    ├──case substr(ID,2,4) = '7251'
550  │    │       store D to t7251d
551  │    │       store LSDATE to t7251s
552  │    │       store LFDATE to t7251f
553  │    ├──case substr(ID,2,4) = '7252'
554  │    │       store D to t7252d
555  │    │       store LSDATE to t7252s
556  │    │       store LFDATE to t7252f
557  │    ├──case substr(ID,2,4) = '7253'
558  │    │       store D to t7253d
559  │    │       store LSDATE to t7253s
560  │    │       store LFDATE to t7253f
561  │    ├──case substr(ID,2,4) = '7254'
562  │    │       store D to t7254d
563  │    │       store LSDATE to t7254s
564  │    │       store LFDATE to t7254f
565  │    └──endcase
566  │       skip
567  └──enddo
568     *****************************************************************************
569     @ x,y+10  say pFont1+U_on+'Block Number : '+temp8lk+U_off+pFont4
570     @ x,y+10  say ' '
571     x = x + 2
572     @  x,  2 say U_on+'STEEL'
573     @  x, 17 say 'Start'
574     @  x, 26 say 'Comp'
575     @  x, 35 say 'Dur'+U_off
576     @  x, 35 say ' '
577     @  x, 40 say U_on+'OUTFITTING'
578     @  x, 56 say 'Start'
579     @  x, 65 say 'Comp'
580     @  x, 74 say 'Dur'+U_off
581     @  x, 74 say ' '
582     x = x + 1
583     *****************************************************************************
584     @  x,  8 say 'Fab(711)'
585  ┌──if t711d > 0
586  │     @  x, 17 say t711s
587  │     @  x, 26 say t711f
588  │     @  x, 35 say t711d picture '999'
589  └──endif
590     @  x, 41 say 'O/F Inv.(7251)'
591  ┌──if t7251d > 0
592  │     @  x, 56 say t7251s
593  │     @  x, 65 say t7251f
594  │     @  x, 74 say t7251d picture '999'
595  └──endif
596     x = x + 1
597     @  x,  3 say 'Sub-Assy(712)'
598  ┌──if t712d > 0
```

loads dates to variables

prints headings for date por of the Gantt Schedule

```
599         @  x, 17 say t712s
600         @  x, 26 say t712f
601         @  x, 35 say t712d picture '999'
602      endif
603      @  x, 41 say 'O/F Up-R(7252)'
604      if t7252d > 0
605         @  x, 56 say t7252s
606         @  x, 65 say t7252f
607         @  x, 74 say t7252d picture '999'
608      endif
609      x = x + 1
610      @  x,  3 say 'Assembly(713)'
611      if t713d > 0
612         @  x, 17 say t713s
613         @  x, 26 say t713f
614         @  x, 35 say t713d picture '999'
615      endif
616      @  x, 42 say 'S/B & P(7253)'
617      if t7253d > 0
618         @  x, 56 say t7253s
619         @  x, 65 say t7253f
620         @  x, 74 say t7253d picture '999'
621      endif
622      x = x + 1
623      @  x,  3 say 'Stacking(715)'
624      if t715d > 0
625         @  x, 17 say t715s
626         @  x, 26 say t715f
627         @  x, 35 say t715d picture '999'
628      endif
629      @  x, 41 say 'O/F Post(7254)'
630      if t7254d > 0
631         @  x, 56 say t7254s
632         @  x, 65 say t7254f
633         @  x, 74 say t7254d picture '999'
634      endif
635      x = x + 2
636      ********************************************************************************
637
638   <===return
639
640
641
642      ***************************
643      *   PROCEDURE G_PAGE_G    *
644      *    for Grand Blocks     *
645      ***************************
646
647      PROC G_PAGE_G
648
649      ********************************************************************************
650      go top
651      store ' ' to tHEAD1
652      store ' ' to tHEAD2
653      store ' ' to tHEAD3
654      store ' ' to tSHADE1
655      store ctod(' ') to S_date
656      store LSDATE-20 to WK_Start
657      do DATE_BAR
658      store 1  to x
659      store 45 to y
660      @ x,y-35  say pFont2+'Grand Block '+C4+' Build Strategy'+pFont
661      @ x,y-35  say ' '
662      @ x,y     say p8lpi
663      x = x + 2
664      @ x,y-32  say chr(201)
665      @ x,y-31  say replicate(chr(205),100)
666      @ x,y+69  say chr(187)
667      x = x + 1
668      @ x,y-32  say chr(186)
669      @ x,y     say tHEAD1
670      @ x,y+69  say chr(186)
671      x = x + 1
672      @ x,y-32  say chr(186)
673      @ x,y     say tHEAD2
```

print schedule dates

prints date bar headings for Gantt portion of the Gantt Schedu

```
674     @ x,y+69  say chr(186)
675     x = x + 1
676     @ x,y-32  say chr(186)
677     @ x,y     say tHEAD3
678     @ x,y+69  say chr(186)
679     x = x + 1
680     ********************************************************************************
681  ┌──do while .not. eof()
682  │      store substr(ID,6,3) to temp8lk
683  │      @ x,y-32  say chr(186)
684  │      @ x,y     say tSHADE1
685  │      @ x,y+69  say chr(186)
686  │      x = x + 1
687  │      @ x,y-32  say chr(186)
688  │      @ x,y     say tSHADE1
689  │      @ x,y-25  say U_on+pFont1+'Block '+temp8lk+U_off+pFont
690  │      @ x,y-25  say ' '
691  │      @ x,y+69  say chr(186)
692  │      x = x + 1
693  │   ┌──do while substr(ID,6,3) = temp8lk
694  │   │     ┌──if D > 0
695  │   │     │      store ' ' to tErect
696  │   │     │      @ x,y     say tSHADE1
697  │   │     │   ┌──do case
698  │   │     │   ┌──case substr(ID,2,4) = '7110'
699  │   │     │   │      store '"Stl"          Fabrication' to tDescrpt
700  │   │     │   │      store chr(220) to tGantt
701  │   │     │   ┌──case substr(ID,2,4) = '7120'
702  │   │     │   │      store '"Stl"         Sub-Assembly' to tDescrpt
703  │   │     │   │      store chr(220) to tGantt
704  │   │     │   ┌──case substr(ID,2,4) = '7130'
705  │   │     │   │      store '"Stl"             Assembly' to tDescrpt
706  │   │     │   │      store chr(220) to tGantt
707  │   │     │   ┌──case substr(ID,2,4) = '7251'
708  │   │     │   │      store '"O/F" Pre-Blast (Inverted)' to tDescrpt
709  │   │     │   │      store chr(207) to tGantt
710  │   │     │   ┌──case substr(ID,2,4) = '7252'
711  │   │     │   │      store '"O/F" Pre-Blast (Up-Right)' to tDescrpt
712  │   │     │   │      store chr(209) to tGantt
713  │   │     │   ┌──case substr(ID,2,4) = '7253'
714  │   │     │   │      store '"O/F"       Blast & Paint' to tDescrpt
715  │   │     │   │      store chr(127) to tGantt
716  │   │     │   ┌──case substr(ID,2,4) = '7254'
717  │   │     │   │      store '"O/F"           Post Blast' to tDescrpt
718  │   │     │   │      store chr(223) to tGantt
719  │   │     │   ┌──case substr(ID,2,4) = '7150'
720  │   │     │   │      store '"Stl"             Stacking' to tDescrpt
721  │   │     │   │      store chr(177) to tGantt
722  │   │     │   │      store chr(004) to tE_Date
723  │   │     │   │      store 'Y'      to tErect
724  │   │     │   ┌──case substr(ID,2,4) = '7160'
725  │   │     │   │      store '"Stl"             Erection' to tDescrpt
726  │   │     │   │      store chr(177) to tGantt
727  │   │     │   │      store chr(004) to tE_Date
728  │   │     │   │      store 'Y'      to tErect
729  │   │     │   └──endcase
730  │   │     │   ┌──if tErect = ' '
731  │   │     │   │      @ x,y-32  say chr(186)
732  │   │     │   │      @ x,y-30  say tDescrpt
733  │   │     │   │      @ x,y+(((LSDATE-S_date)/7)  ) say replicate (tGantt,int((LFDATE-LSDATE)/7)+1)
734  │   │     │   │      @ x,y+(((LSDATE-S_date)/7)-9) say LSDATE
735  │   │     │   │      @ x,y+(((LSDATE-S_date)/7)+int((LFDATE-LSDATE)/7)+2) say LFDATE
736  │   │     │   │      @ x,y+69  say chr(186)
737  │   │     │   │      x = x + 1
738  │   │     │   ┌──else
739  │   │     │   │      @ x,y-32  say chr(186)
740  │   │     │   │      @ x,y-30  say tDescrpt
741  │   │     │   │      @ x,y+(((LSDATE-S_date)/7)  ) say '*'
742  │   │     │   │      @ x,y+(((LSDATE-S_date)/7)  ) say replicate (tGantt,int((LFDATE-LSDATE)/7)+1)
743  │   │     │   │      @ x,y+(((LSDATE-S_date)/7)-9) say LSDATE
744  │   │     │   │      @ x,y+(((LSDATE-S_date)/7)+int((LFDATE-LSDATE)/7)+2) say LFDATE
745  │   │     │   │      @ x,y+69  say chr(186)
746  │   │     │   │      x = x + 1
747  │   │     │   └──endif
748  │   │     └──endif
```

prints Gantt bars for the C
portion of the schedule

```
749         skip
750       enddo
751     enddo
752     @ x,y-32  say chr(200)
753     @ x,y-31  say replicate(chr(205),100)
754     @ x,y+69  say chr(188)
755     ********************************************************************************
756
757 <===return
758
759
760
761
762     ************************************************
763     *           PROCEDURE DATE_BAR                 *
764     *  Creates the Date Bar Heading Variables      *
765     *        used in the Gantt Schedules           *
766     ************************************************
767
768     PROC DATE_BAR
769
770   do case
771   case month(WK_Start) = 1
772       store str(year(WK_Start)-1,4) to tYR
773       store ctod('11/01/'+substr(tYR,3,2)) to S_date
774       store 'Nov Dec  Jan Feb Mar  Apr May Jun  Jul Aug Sep  Oct Nov Dec  Jan Feb' to tHEAD1
775       store '4     5  0        1        2        3        4        5  0     ' to tHEAD2
776       store '456789012123456789012345678901234567890123456789012345678901234567' to tHEAD3
777       store b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s3 to tSHADE1
778   case month(WK_Start) = 2
779       store str(year(WK_Start)-1,4) to tYR
780       store ctod('12/01/'+substr(tYR,3,2)) to S_date
781       store 'Dec  Jan Feb Mar  Apr May Jun  Jul Aug Sep  Oct Nov Dec  Jan Feb Mar' to tHEAD1
782       store '  5  0       1        2        3        4        5  0        1 ' to tHEAD2
783       store '8901212345678901234567890123456789012345678901234567890121234567890' to tHEAD3
784       store s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b3 to tSHADE1
785   case month(WK_Start) = 3
786       store str(year(WK_Start),4) to tYR
787       store ctod('01/01/'+substr(tYR,3,2)) to S_date
788       store 'Jan Feb Mar  Apr May Jun  Jul Aug Sep  Oct Nov Dec  Jan Feb Mar  Apr' to tHEAD1
789       store '0        1        2        3        4        5  0        1      ' to tHEAD2
790       store '12345678901234567890123456789012345678901234567890121234567890123456' to tHEAD3
791       store b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s3 to tSHADE1
792   case month(WK_Start) = 4
793       store str(year(WK_Start),4) to tYR
794       store ctod('02/01/'+substr(tYR,3,2)) to S_date
795       store 'Feb Mar  Apr May Jun  Jul Aug Sep  Oct Nov Dec  Jan Feb Mar  Apr May' to tHEAD1
796       store '0    1        2        3        4        5  0        1        2' to tHEAD2
797       store '56789012345678901234567890123456789012345678901212345678901234567890' to tHEAD3
798       store s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b3 to tSHADE1
799   case month(WK_Start) = 5
800       store str(year(WK_Start),4) to tYR
801       store ctod('03/01/'+substr(tYR,3,2)) to S_date
802       store 'Mar  Apr May Jun  Jul Aug Sep  Oct Nov Dec  Jan Feb Mar  Apr May Jun' to tHEAD1
803       store ' 1        2        3        4        5  0        1        2    ' to tHEAD2
804       store '90123456789012345678901234567890123456789012123456789012345678901234' to tHEAD3
805       store b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s3 to tSHADE1
806   case month(WK_Start) = 6
807       store str(year(WK_Start),4) to tYR
808       store ctod('04/01/'+substr(tYR,3,2)) to S_date
809       store 'Apr May Jun  Jul Aug Sep  Oct Nov Dec  Jan Feb Mar  Apr May Jun  Jul' to tHEAD1
810       store '1     2        3        4        5  0        1        2         ' to tHEAD2
811       store '45678901234567890123456789012345678901212345678901234567890123456789' to tHEAD3
812       store s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b3 to tSHADE1
813   case month(WK_Start) = 7
814       store str(year(WK_Start),4) to tYR
815       store ctod('05/01/'+substr(tYR,3,2)) to S_date
816       store 'May Jun  Jul Aug Sep  Oct Nov Dec  Jan Feb Mar  Apr May Jun  Jul Aug' to tHEAD1
817       '  2        3        4        5  0        1        2        3    ' to tHEAD2
818       store '8901234567890123456789012345678901212345678901234567890123456789012' to tHEAD3
819       store b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s3 to tSHADE1
820   case month(WK_Start) = 8
821       store str(year(WK_Start),4) to tYR
822       store ctod('06/01/'+substr(tYR,3,2)) to S_date
823       store 'Jun  Jul Aug Sep  Oct Nov Dec  Jan Feb Mar  Apr May Jun  Jul Aug Sep' to tHEAD1
```

```
824         store '2       3         4         5  0        1         2         3       ' to tHEAD2
825         store '234567890123456789012345678901212345678901234567890123456789012345678901234567' to tHEAD3
826         store s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b3 to tSHADE1
827      case month(WK_Start) = 9
828         store str(year(WK_Start),4) to tYR
829         store ctod('07/01/'+substr(tYR,3,2)) to S_date
830         store 'Jul Aug Sep  Oct Nov Dec  Jan Feb Mar  Apr May Jun  Jul Aug Sep  Oct' to tHEAD1
831         store '   3         4         5  0        1         2         3         4   ' to tHEAD2
832         store '78901234567890123456789012123456789012345678901234567890123456789012' to tHEAD3
833         store b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s3 to tSHADE1
834      case month(WK_Start) = 10
835         store str(year(WK_Start),4) to tYR
836         store ctod('08/01/'+substr(tYR,3,2)) to S_date
837         store 'Aug Sep  Oct Nov Dec  Jan Feb Mar  Apr May Jun  Jul Aug Sep  Oct Nov' to tHEAD1
838         store '3         4         5  0        1         2         3         4      ' to tHEAD2
839         store '12345678901234567890121234567890123456789012345678901234567890123456' to tHEAD3
840         store s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b3 to tSHADE1
841      case month(WK_Start) = 11
842         store str(year(WK_Start),4) to tYR
843         store ctod('09/01/'+substr(tYR,3,2)) to S_date
844         store 'Sep  Oct Nov Dec  Jan Feb Mar  Apr May Jun  Jul Aug Sep  Oct Nov Dec' to tHEAD1
845         store '    4         5  0        1         2         3         4         5' to tHEAD2
846         store '56789012345678901212345678901234567890123456789012345678901234567890' to tHEAD3
847         store b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s3 to tSHADE1
848      case month(WK_Start) = 12
849         store str(year(WK_Start),4) to tYR
850         store ctod('10/01/'+substr(tYR,3,2)) to S_date
851         store 'Oct Nov Dec  Jan Feb Mar  Apr May Jun  Jul Aug Sep  Oct Nov Dec  Jan' to tHEAD1
852         store '4         5  0        1         2         3         4         5  0  ' to tHEAD2
853         store '01234567890121234567890123456789012345678901234567890123456789012123' to tHEAD3
854         store s4+b4+s5+b4+s4+b5+s4+b4+s5+b4+s4+b5+s4+b4+s5+b3 to tSHADE1
855      endcase
856
857 <===return
858
859
860     *************************
861     * PROCEDURE SCREEN1     *
862     *************************
863
864     PROC SCREEN1
865
866     ********************************************************************************
867     set color to gr+/b
868     @ 1,0 to 22,79 double
869     @ 2,  1 clear to  3,78
870     @ 2,  6 say 'Block No.'
871     @ 2, 22 say 'Contract'
872     set color to w+/b
873     @ 2, 16 say tBlock
874     @ 2, 31 say tContract
875     @ 2, 34 say Hullname
876     set color to w+/r
877     @ 4,  1 clear to 21,78
878     set color to gr+/r
879     @ 4,  2 say 'STEEL'
880     set color to w+/r
881     @ 4, 17 say 'Start'
882     @ 4, 26 say 'Comp'
883     @ 4, 35 say 'Dur'
884     set color to gr+/r
885     @ 4, 40 say 'OUTFITTING'
886     set color to w+/r
887     @ 4, 56 say 'Start'
888     @ 4, 65 say 'Comp'
889     @ 4, 74 say 'Dur'
890     ********************************************************************************
891
892     set color to w+/r
893     @ 5,  8 say 'Fab(711)'
894     set color to w/r
895     if t711d > 0
896        @ 5, 17 say t711s
897        @ 5, 26 say t711f
898        @ 5, 35 say t711d picture '999'
```

displays date headings for the date section of the Gantt sche

```
       ~ i f
      set color to */r
      a 5, 41 say '0/F Inv. (7251)'
      0000000000000000000000000000set color to w/r
      if t7251d > 0
          a 5, 56 say t7251s
          a 5, 65 say t7251f
          a 5, 74 say t7251d picture '999'
  L       i f

      set color to m+/r
      a 6, 3 say 'sub-Assy(712)'
      set color to w/r
      if t712d > 0
          a 6, 17 say t712s
          a 6, 26 say t712f
          a 6, 35 say t712d picture '999'
  L       i f
      set color to uw
      a 6, 41 say '0/F UP-R(7252)'
      set color to u/r
  r   if t7252d > 0
          a 6, 56 say t7252s
          a 6, 65 say t7252f
          a 6, 74 say t7252d picture 19991
  L       i f

      set color to n+lr
      a 7, 3 say 1Asss41Y(713) '
      set color to "W/r
  |—if t713d > 0
  |       a 7, 17 say t713s
  |       a 7, 26 say t713f
  |       a 7, 35 say t713d picture 19991
  |-endif
      set color to w+/r
      a 7, 42 say 'S/B & P(7253)'
      set color to w/r
      if t7253d > 0
          a 7, 56 say 000000000000000000000000000000000000000t7253s
0000000000000070,0650000y0t7253f00000000000000000000000000000000000000000000000000000000000000000000000000000000
          a 7, 74 say t7253d picture '9990
  L       i f

  |—if tstack = ' '
  |       set color to W/r
  |       a 8, 3 say 'Erection(716)'
  |       set color to Wlr
  |       if t716d > 0
  |           a 8, 17 say t716s
  |           a 8, 26 say t716f
  |           a 8, 35 say t716d picture '999'
  |   L       i f
  |-else
  |       set color to w+/r
  |       a 8, 3 say 'Stacking(715)'
  |       set color to hi/r
  |       if t715d > 0
  |           a 8, 17 say t715s
  |           a 8, 26 say t715f
  |           a 8, 35 say t715d picture '999'
  |   L       i f
  |-endif
      set color to W+/r
      a 8, 41 say 80/F Post(7254)'
      set color to W/r
      if t7254d > 0
          a 8, 56 say t7254s
          a 8, 65 say t7254f
          a 8, 74 say t7254d picture '999'
  L       i f
      ****************************************************************************

      set color to W +/bg
      a 11, 1 clear to 21,78
```

– displays the schedule dates for the date section of the Gantt Schedule

```
974       a 11, 11 say tHEADl
975       a 12, 11 say tHEAD2
976       a 13, 11 say tHEAD3
977       set color to /bg
978       a 14, 3 say '711'
979       a 15, 3 say '712'
980       a 16, 3 say '713'
981       a 17, 3 say l7251O
982       a 18, 3 say '7252'
983       a 19, 3 say '7253'
984       a 20, 3 say '7254'
985       if tStack = 41
986           a 21, 3 say '716'
987        else
988           a 21, 3 say '715'
989 L     endi f
990
991       if t711d > 0
992           store int((t711s-S_date)/7)+12 to tY1
993           store int((t711f-t711s)/7)+2 to dY1
994           store int((t711s-S_date)/7)+1 to tY2
              set color to w+/g
996           a 14, tY1 say replicate ('-',int((t711f-t711s)/7)+1)
997           set color to /bg
998           a 14, tY2 say t711s
              a 14, tYl+wl say t711f
1000 L    endi f
1001
1002      i f t712d > 0
1003          store int( (t712s-S_dste)/7 )+12 to tYl
1004          store int( (t712f -t712s)/7)+2 to dYl
1005          store intC(t712s-s_dat.&7 )+l to tY2
              set color to u+/g
; r 7         a 15, tYl say replicate ('-!, int((t712f-t712s) /7)+1)
1008          set color to /bg
1009          a 15, tY2 say t712s
1010 r        a 15, tYl+dYl say t712f
1 0 1 1     - i f
1012
1013      if t713d > 0
1014          store int((t713s-S_date)/7)+12 to tYl
1015          store int((t713f-t713 s)/7)+2 to dy1
1016          store int((t713s-S_date)/7 )+1 to tY2
1017          set color to W+/g
1018          a 16, tY1 say replicate ('-',int((t713f-t713s)/7)+1)
1019          set color to /bg
1020          a 16, tY2 say t713s
1021          a 16, tYl+dYl say t713f
1022 [       i f
1023
1024      if t7251d > 0
1025          store int((t~ls-S_date) /7)+12 to tYl
1026          store int( (t7251f -t7251s)/7)+2 to dyl
1027          store int((t7251s-S_dete) /7)+1 to tY2
1028          set color to w+/g
1029          a 17, tYl say repicate ( '*', int((t7251f-t7251 s)/7)+1 )
1030          set color to /bg
1031          a 17, tY2 say t251s
1032          a 17, tY1+dY1 say t7251f
1033 L       i f
1034
1035      if t7252d > 0
1036          store int((t7252s-S_dete) /7)+12 to tYl
1037          store int((t7252f-t7252s) /7)+2 to CM
1038          store int((t7252s-S_date) /7)+1 to tY2
1039          set color to w+/9
1040          i3 18, tYl say replicate ( '*', int((t7252f-t7252s) /7)+1 )
1041          set color to /bg
1042          a 18, tY2 say t7252s
1043          a 18, tYl+dYl say t7252f
1044 L       i f
1045
1046      if t7253d > 0
1047          store int((t7253s-S_date)/7) +12 to ty1
1048 r        store int((t7253f-t7253s)/7)+2 to dy1l
```

displays the Gantt bars for
Gantt bar section of the sche

```
1049          store int( (t7253s-S_date) /7)+l to tY2
1050          set color to w/g
1051          a 19, tYl say replicate ('*', int((t7253f-t7253s) /7)+1)
1052          set color to /bg
1053          a 19, tY2 say t7253s
1054          6a 19, tYl+dYl say t7253f
1055       -endif
1056
1057       -if t7254d s O
1058          store int((t7254s-S_date) /7)+12 to tYl
1059          0000store int((t7254f-t7254s) /7)+2 to dy1
1060          store int((t7254s-S_date) /7)+1 to tY2
1061          set color to w+/g
1062          a 20, tY1 say replicate ('*', int((t7254f-t7254s) /7)+1)
1063          set color to /bg
1064          a 20, tY2 say t7254s
1065          a 20, tY1+dYl1sayt7254f
1066       -endif
1067
1068       if tStack = $ j
1069          store 't7Md' to tErectd
1070          store at716st to tErects
1071          store lt716ft to tErectf
1072        else
1073          store 't715d' to tErectd
1074 F        store 't715s' to tErects
1075          store 't715f' to tErectf
1076 L       if
1077       if &tErectd > 0
1078          store int((&tErects-S_date) /7)+12 to tY1
1079          store int((&tErectf-&tErects) /7)+2 to dyl
1080          store int((&tErects-S_date) /7)+1 to tY2
1081          set color to W+/g
1082          a 21, tYl say replicate ('-', int((&tErectf-&tErects) /7)+l)
1083          set color to /bg
1084          a 21, tY2 say &tErects
1085          a 21, tY1+dY1 say &tErectf
1086 L       i f
1087       •                                    ~                              ~
1088
1089 <=== return
1090
1091
1092       *          *          __          ~
1093       *                PROCEOURE HENU-1          *
1094       •    Blank for Gantt Hew Format *
1095       *                *                ~
1096
1097       PROC MENU-1
1098
1099       set color to gr+/b
1100       a 4, 17 to 18, 58 double
1101       a 4, 30 SAY "GANTT CHART MENU !!
1102       set color to u+/r
1103       a 5, 18 SAY '
1104       a 6, 18 SAY '
1105       a 7, 18 SAY '
1106       a 8, 18 SAY '
1107       a 9, 18 SAY '
1108       a 10, 18 SAY '
1109       a 11, 18 SAY '
1110       a 12, 18 SAY '
1111       a 13, 18 SAY '
1112       a 14, 18 SAY '
1113       a 15, 18 SAY '
1114       a 16, 18 SAY '
1115       a 17, 18 SAY '
1116       set color to gr
1117       a 19, 26 say 'Press l!Esc" to Return'
1118
1119 <===return
1120
1121
1122       **********************************
1123       •          PROCEOURE    UENU_2    *
```

```
1124      *    Text for Gantt Hem Format •
1125      **********************************
1126
1127      PROC MENU_2
1128
1129      set color to gr+/b
1130      a 4, 17 to 18, 58 double
1131
1132      set color to gr+/rl
1133      a 5, 18 SAY '
1134      a 6, 18 SAY '
1135      3 7, 18 SAY 8       Enter Contract Letter : :
1136      3 8, 18 SAY @       Enter Block No. : :
1137      3 9, 18 SAY '
1138      3 10, 18 SAY '
1139      set color to w/r
1140      3 11, 18 SAY '      Contrsct Letters
1141      3 12, 18 SAY '         llxm = W/V UELL PLANNED
1142      3 13, 18 SAY '
1143      3 14, 18 SAY s
1144      3 15, 18 SAY '
1145      3 16, 18 SAY '
1146      3 17, 18 SAY '
1147      set color to gr
114s      a 19, 26 say 'Press "Esc" to Return'
1149
1150  <===return
1151
1152
1153      ● Formtted by: dANALYST Ver. 7.3a on March 24, 1992 at 9:48 AM
dANALYST found O error(s), O warnin9(S), 153 lines.
```

```
INPUT FILE: C:\OPLAN\SP8_LAY .PRG
    1       *  FILE NAME: SP8_LAY.PRG
            *  BY: D. HcQuaide
    3       *  DATE: March 24, 1992
    4       *  DESC:
    5       *  CALLED BY:
    6       *  DATA FILES:
    7       * SP8-LAY. prg
    8
            set procedure  to SP8_LAY
   10
   11       *                                                              ~
   12       store space(30) to tModel
   13       set color to gr+/b
   14       a 4, 17 to 18, 58 double
   15       a 4, 30 SAY " LAYOWN SCHEDULE u
   16       set color to w+/r
   17       a 5, 18 SAY 8                                          8
   18       a 6, 18 SAY '                                          1
   19       a 7, 18 SAY 1       Enter klodel Ume w/Path            1
   20       a 8, 18 SAY :                                          1
  201       a 9, 18 SAY 8                                          I
   22       a 10, 18 SAY 1                                         I
   23       all, 18 SAY '                                          1
   24       a 12, 18 SAY 6                                         I
   25       a 13, 18 SAY 8                                         [
   26       a 14, 18 SAY m                                         1
   27       a 15, 18 SAY a                                         1
   28       a 16, 18 SAY 8                                         I
            a 17, 18 say 8                                         1
   30       set color to gr
   31       a 19, 26 say 'Press "Esc" to Return'
   32          while .T.
   33           a 9, 23 get tModel  picture  !!!!! !!!! !!!!! !!!!! !!!!!!!!!!°
   34 ┌         reed
   35 │      ┌──if readkey() = 12
   36 <===<===<===return
   37 │      └──endif
   38 │
   39 │         store trim(tModel)+'.act' to dModel
   40 │      ┌──if file(dModel)
   41 <===<===<===exit
   42 │           else
   43 │              set color to bt+/r
   44 │              a 15, 18 SAY '                File Does Not Exist
   45 │              set color to H/r
   46 │              i) 15, 26 SAY °Activity'
   47 │              set color to w/r
   50 └──enddo
   ii       *                    ~                  —                  —
   52       select A
   53       use Model index TABLE_LS alias MODEL
   54       set color to w*/r
   55       a 15, 18 SAY [       Packing & Reirx&exing File         I
   56       set color to n+lr
   57       a 7, 1 8 s A Y #                                       8
   58       a 8,18 SAY'                                            1
   59       a 9, 1 8 s A Y '                                       I
   60       a 15, 18 SAY 1                                         I
   61       peck
   62       reindex
   63
   64       *                                                              ~
   65
   66       *- setting @ the Print CodeS **
   67       * "normal_1"   Standard Setup, used to print lines,background,gantt
   68       store chr(027)+'(8Q'+chr(027)+'(s0p16.67h8.5v0s0b3T'+chr(027)+'&l10' to normal_1
   69       * "normal_P"   Standard Setup, used to print lines,background,gantt
   70       store chr(027)+'(8Q'+chr(027)+'(s0p16.67h8.4v0s0b3T'+chr(027)+'&l00' to normal_P
   71       * "normal_2"   used to print Data Information
   72       store chr(027)+'(0U'+chr(027)+'(s0p16.67h9.6v0s0b6T'+chr(027)+'&l10' to normal_2
   73       * "resetP"   Reset Printer ***                                                I
```

read and verify model name and files

set up data files

```
   store chr(027)+'E' to resetP
   * "legel" Set for legel paper ***
   store chr(027)+'&l3A' to legel
   * "PRT_lpi6" 6 Lines per Inch ***
   store chr(027)+'&l6D' to PRT_lpi6
   * "PRT_lpi8" 8 Lines per Inch ***
   store chr(027)+'&l8D' to PRT_lpi8
   * "under_y"   Starts underlining print ***
   store chr(027)+'&dD' to under_y
   * "under_n"   Stops underlining print ***
   store chr(027)+'&d@' to under_n
   * "BOLD_1"   used to print Titles
   store chr(027)+'(0U'+chr(027)+'(s0p10.00h13.9v0s3b11T' to BOLD_1
   * "BOLD_2"   used to print Titles
   store chr(027)+'(0U'+chr(027)+'(s0p8.11h16.1v0s3b11T' to BOLD_2
   * "BOLD_3"   used to print Titles
   store chr(027)+'(0U'+chr(027)+'(s0p6.53h18.0v0s3b11T' to BOLD_3
   ********************************************************************************

   * Re-Calculate the Gantt Placements for print-out **
   set color to w+/r,w+/r
   store 'Y' to tPLACE
   @  7, 18  say ' Recalculating Gantt Placements (Y/N)   '
   @  7,56 get tPLACE picture '!'
   read
   if readkey() = 12
<===<===return
   endif
   if tPLACE = 'Y'
      set color to w+/r,w+/r
      @ 14, 18  say '                    Gantt Placements     '
      set color to w+*/r
      @ 14, 23  say 'Recalculating'
      do PLACEMNT
      set color to w+/r
      @ 14, 18  say '                                         '
   endif
   ********************************************************************************

   do while .T.
      store ' ' to t92A,t92B,t93A,t93B
      store ' ' to tShop,tTable1,tTable2
      store ' ' to tArea_A,tArea_B
      set color to gr+/r, /w
      @  5, 18  SAY '  Steel               Outfitting        '
      set color to  w+/r
      @  7, 18  SAY '      Shop                Area "A"      '
      @  8, 18  SAY '      Table #1            Area "B"      '
      @  9, 18  SAY '      Table #2                          '
      set color to gr+/r
      @ 12, 18  SAY '                  1992 1993             '
      set color to  w+/r
      @ 13, 18  SAY '       Jan->Jun                         '
      @ 14, 18  SAY '       Jul->Dec                         '
      set color to gr+/r
      @ 16, 18  SAY '    Select with an "X" Your Choices     '
      @  7, 22  get tShop   picture '!'
      @  8, 22  get tTable1 picture '!'
      @  9, 22  get tTable2 picture '!'
      @  7, 42  get tArea_A picture '!'
      @  8, 42  get tArea_B picture '!'
      @ 13, 37  get t92A    picture '!'
      @ 14, 37  get t92B    picture '!'
      @ 13, 42  get t93A    picture '!'
      @ 14, 42  get t93B    picture '!'
      read
      if readkey() = 12
<===<===<===return
      endif
      ********************************************************************************

      set color to  w+/r
      @ 16, 18  SAY '      Setting up Memory Variables      '

      *** Memory Variables to Create Calendar Grid ***
```

Set up print variables

verify need to re-calculate table placements

read tables and time spans for print outs

```
149         store         '+chr(177)+chr(177)   to linewl
150         store ' ' to dl
151         store CHR(176) TO d2
152         store CHR(177) TO d3
153         store CHR(177)+CHR(17'7) TO d4
154         store dl+d2+dl+d2+dl+d4 TO wk
155         store d3td2+dl+d2+dl+d4 TO wkm
156         store dl+d3+dl+d2+dl+d4 TO wkt
157         store dl+d2+d3+d2+dl+d4 TO wku
158         store dl+@dl+cf3+dl+d4 TO wkth
159         store dl+d2+dl+d2+d3+& TO wkf
            store                 TO        wkmas
            store dl+d2+dl+d3+d3* TO wkthank
162         store cEWB+dl+d2+dl+d4 TO wkmt
163         store d3+d3+dM@+dl+d4 TO wkmtu
164
165          Start Dates of each Grid
166         store ctod( @ 12/30/91 ) to start92a
167         store ctod( t 06/29/92 ) to start92b
168         store ctod(812/28/92~ ) to start93a
169         store ctod( '06/28/93') to start93b
170         *
171
172         *** Calendar Headings & Grid for First Half of 1992 ***
173         store ' |          JANUARY 1992          |          FEBRUARY 1992          |          MARCH 1992          |          APRIL 1992
174         store '33          1111111111222222222233          1111111112222222222          1111111112222222222233          1111111111
175         store '0112345678901234567890123456789011234567890123456789012345678901234567890123456789012345678901234567890123456789
176         store '   1       2       3       4       5       6       7       8       9       10      11      12      13      14      15      16
177         store wku+wk+wk+wk+wk+wk+wk+wk+wk+wk+wk+wk+wk+wk+wk+wk+wkf+wk+wk+wk+wk+wk+wkm+wk+wk+wk+wk to dtln_E92A
178
179         *** Calendar Headings & Grid for Second Half of 1992 ***
180         store ' |          JULY 1992              |          AUGUST 1992            |          SEPTEMBER 1992         |          OCTOBER 19
181         store '23          1111111111222222222233          1111111111222222222233          1111111111222222222223          111111111
182         store '9012345678901234567890123456789011234567890123456789012345678901234567890123456789012345678901234567890123456789012345678
183         store '  27      28      29      30      31      32      33      34      35      36      37      38      39      40      41      42
184         store wkf+wk+wk+wk+wk+wk+wk+wk+wk+wk+wkm+wk+wk+wk+wk+wk+wk+wk+wk+wk+wkthank+wk+wk+wk+wkthank to dtln_E92B
185
186         *** Calendar Headings & Grid for First Half of 1992 ***
187         store '   |          JANUARY 1993           |          FEBRUARY 1993          |          MARCH 1993          |          APRIL 199
188         store '2233          1111111111222222222233          1111111112222222222          1111111112222222222233          111111111
189         store '8901123456789012345678901234567890112345678901234567890123456781234567890123456789012345678901123456789012345678
190         store '   1       2       3       4       5       6       7       8       9       10      11      12      13      14      15      16
191         store wkxmas+wk+wk+wk+wk+wk+wk+wk+wk+wk+wk+wk+wk+wk+wkf+wk+wk+wk+wk+wk+wk+wkm+wk+wk+wk to dtln_E93A
192
193         *** Calendar Headings & Grid for Second Half of 1992 ***
194         store '   |          JULY 1993              |          AUGUST 1993            |          SEPTEMBER 1993         |          OCTOBER 1
195         store '223          1111111111222222222233          1111111111222222222233          1111111111222222222223          11111111
196         store '8901234567890123456789012345678901123456789012345678901234567890112345678901234567890123456789012345678901234567
197         store '  27      28      29      30      31      32      33      34      35      36      37      38      39      40      41      42
198         store wkf+wk+wk+wk+wk+wk+wk+wk+wk+wk+wkm+wk+wk+wk+wk+wk+wk+wk+wk+wk+wkthank+wk+wk+wk+wk to dtln_E93B
            ********************************************************************************
201         store "PLACEMENT = ' 1'" to tCondit1
202         store "PLACEMENT = ' 2'" to tCondit2
203         store "PLACEMENT = ' 3'" to tCondit3
204         store "PLACEMENT = ' 4'" to tCondit4
            store "PLACEMENT = ' 5'" to tCondit5
            store "PLACEMENT = ' 6'" to tCondit6
207         store "PLACEMENT = ' 7'" to tCondit7
208         store "PLACEMENT = ' 8'" to tCondit8
209         store "PLACEMENT = ' 9'" to tCondit9
210         store "PLACEMENT = '10'" to tcondit10
211         ********************************************************************************
212
213         set color to  w+*/r
214         @ 16, 18  SAY '                Printing                '
215
216       —if tShop = 'X'
217            store 'PLATEN LAYDOWN SCHEDULE' TO title1
218            store 'SHOP ASSEMBLY' to title2
219            set color to w+*/r
220            @  7, 24  SAY 'Shop'
221            store 'SHOP' to body
222            do PRT_LAY
223            set color to w+/r
```

set up background variables

Set up date bar heading variables

Set up print condition variables

```
224         &l 7, 24 SAY 'shop'
225     endif
226
227     if table = 'X'
228         store 'PLATEN LAYDOWN SCHEDULE' TO title
229         store 'TABLE #1' to title2
230         set color to w+/r
231         a 8, 24 SAY 'Table #1'
232         store 'TABLE' to body
233         do PRT_LAY
234         set color to W+/r
235         a 8, 24 SAY lTable #1'
236 [       if
237
238     if tTable2 = 'X'
239         store 'PLATEN LAYDOWN SCHEDULE' TO title
240         store 'TABLE #2° to title2
241         set color to w+*/r
242         a 9, 24 SAY 'Table #2'
243         store 'TABLE2' to body
244         do PRT-LAY
245         set cotor to w+/r
246         a 9, 24 SAY 'Table #2'
247     endif
248
249 —if tArea_A = 'X'
250         store 'ON- BLOCK O/F LAYDOWN SCHEDULE' TO titlel
251         store 'AREA 'A' to title2
252         set color to w+/r
253         a 7, 44 SAY 'Area "A"
254         store '          Printing Table #2          ' to tMEMOl
255         store 'AREA_A' to body
256         do PRT_LAY
257         set color to n+lr
258         a 7, 44 SAY 'Area 'lA'
259     endif
260
261     if  tArea_B = 'X'
262         store 'ON-BLOCK O/F LAYDOWN SCHEDULE' TO titlel
263         store 'AREA 'B' to title2
264         set color to w+*/r
265         a 8, 44 SAY 'Area "B-l
266         store '          Printing Table #2          to tMEHO1
267         store 'AREA-B' to body
268         do PRT_LAY
269         set coior to w+/r
270         a 8, 44 SAY 'Area 'lB'
271     endif
272 —enddo
273 ************************************************************************
274
275 <===retum
276
277
278 ************************************************
279 *             PROCEDURE PRT_LAY                *
    • Prints Time spans for each Table Selected •
281 ************************************************
282
283 PRW PRT_LAY
284
285 store O to x
286 store O to y
287 set device to print
288 a x,y say legel
289 a x,Y say normal-l
290 a x,y say PRT-1P18
291 set device to screen
292
293 —if t92A = 'X'
294         set device to screen
295         set color to 9r+/r
296         a 12, 35 say '1992'
297         set color to w+/r
298         a 13, 25 say 'Jan>Jm
```

- print table laydowns as selected

```
299         set device to print
300         store dtin-A92A to LineA
301         store dtin-W2A to LineB
302         store dtln-C92A to Linec
303         store dtln-D92A to LineD
304         store dtln-E92A to LineE
305         store start92A-1 to tSTART
306         store start92A+182 to tCOMPLETE
307         store        to title3
308         do HEADING
309         do &BODY
310         set device to screen
311         set color to gr+/r
312         a 12, 35 say '1992'
313         set color to W/r
314         a 13, 25 say 'Jan->Jun"
315       endif
316
317       if t92B= 'X'
318         set device to screen
319         set color to 9r+*/r
320         a 12, 35 say '1992
321         set color to u+*/r
322         a 14, 25 say 'Jut->Dec'
323         set device to print
324         store dtln A92B to lineA
325         store dtln-B92B to lineB
326         store dtln-C92B to Linec
327         store dtln-D92B to lind)
328         store dtln-E92B to linsE
329         store start92B 1 to tSTART
330         store start92B+182 to tCOMPLETE
331         store          to title3
332         do HEADING
333         do &BODY
334         set device to screen
335         set color to gr+/r
            a 12, 35 say 81992:
336         set cotor to bf+/r
338         a 14, 25  say 'Jul->Dec'
339       endif
340
341       if t93A= 'X'
342         set device to screen
343         set color to gr+*/r
344         a 12, 40 say '1993'
345         set color to u+*/r
346         a 13, 25 say 'Jan->Jun'
347         set device to print
348         store dtln-A93A to lineA
            store dtln-B93A to lineB
350         store dtln-C93A to linec
351         store dtln-D93A to limo
352         store dtln-E93A to lineE
353         store start93A-1 to tSTART
354         store start93A+lB2 to tCCUPLETE
355         store '         to title3
356         do HEADING
357         do &BODY
358         set device to screen
359         set color to gr+/r
340         a 12, 40 say '1993'
361         set color to w+/r
362         a 13, 25  say 'Jan->Jun'
363       endif
364
365       if t93B = 'X'
366         set device to screen
367         set color to gr+*/r
368         a 12, 40 say '1993'
369         set color to w+*/r
370         a 14, 25  say 'Jul->Dec'
371         set device to print
372         store dtln-A93B to lineA
373         store dth-B93B to lineB
```

```
374         store dtln_C93B to lineC
375         store dtln_D93B to lineD
376         store dtln_E93B to lineE
377         store start93B-1 to tSTART
378         store start93B+182 to tCOMPLETE
379         store '      ' to title3
380         do HEADING
381         do &BODY
382         set device to screen
383         set color to gr+/r
384         @ 12, 40  say '1993'
385         set color to w+/r
386         @ 14, 25  say 'Jul->Dec'
387      endif
388
389 <===return
390
391
392     ***********************************************
393     *             PROCEDURE HEADING               *
394     *      Prints Titles & Date Bar Headings      *
395     ***********************************************
396
397     PROC HEADING
398
399     x = 1
400     @ x,y+27  say BOLD_2+title1+normal_1
401     @ x,y+27  say ' '
402     @ x,y+90  say BOLD_3+title2+normal_1
403     @ x,y+90  say ' '
404     x = x+1
405     @ x,y     say chr(201)
406     @ x,y+1   say replicate(chr(205),20)
407     @ x,y+21  say chr(203)
408     @ x,y+22  say replicate(chr(205),148)
409     @ x,y+171 say 'Printed on '+dtoc(date())
410     @ x,y+191 say replicate(chr(205),13)
411     @ x,y+204 say chr(187)
412     x = x+1
413     @ x,y     say chr(186)
414     @ x,y+10  say 'MONTH/YEAR'
415     @ x,y+21  say chr(186)
416     @ x,y+22  say lineA
417     @ x,y+204 say chr(186)
418     x = x+1
419     @ x,y     say chr(186)
420     @ x,y+10  say '   WEEK NO'
421     @ x,y+21  say chr(186)
422     @ x,y+22  say lineD
423     @ x,y+22  say replicate(lineW,26)
424     @ x,y+204 say chr(186)
425     x = x+1
426     @ x,y     say chr(186)
427     @ x,y+21  say chr(186)
428     @ x,y+22  say lineB
429     @ x,y+22  say replicate(lineW,26)
430     @ x,y+204 say chr(186)
431     x = x+1
432     @ x,y     say chr(186)
433     @ x,y+10  say '       DAY'
434     @ x,y+21  say chr(186)
435     @ x,y+22  say lineC
436     @ x,y+22  say replicate(lineW,26)
437     @ x,y+204 say chr(186)
438     x = x+1
439     *@ x,y     say chr(199)
440     *@ x,y+1   say replicate(chr(196),20)
441     *@ x,y+21  say chr(215)
442     *@ x,y+22  say replicate(chr(196),182)
443     *@ x,y+22  say lineE
444     *@ x,y+204 say chr(182)
445     *x = x+1
446
447
448 <===return
```

```
449
450
451     ************************************************
452     *              PROCEDURE PLACEMNT              *
453     *     Calculates Table Position Placements     *
454     ************************************************
455
456     PROC PLACEMNT
457
458     go top
459     locate for .not. table = '     '
460 ┌──do while .not. eof()
461 │       store TABLE to tTABLE
462 │       store ' 1' to tPLACE
463 │   ┌──do case
464 │   ├──case TABLE = 'SH'
465 │   │     ┌──do while TABLE = tTABLE
466 │   │     │     ┌──do case
467 │   │     │     ├──case tPLACE = ' 1'
468 │   │     │     │        replace PLACEMENT with ' 1'
469 │   │     │     │        store ' 2' to tPLACE
470 │   │     │     ├──case tPLACE = ' 2'
471 │   │     │     │        replace PLACEMENT with ' 2'
472 │   │     │     │        store ' 3' to tPLACE
473 │   │     │     ├──case tPLACE = ' 3'
474 │   │     │     │        replace PLACEMENT with ' 3'
475 │   │     │     │        store ' 4' to tPLACE
476 │   │     │     ├──case tPLACE = ' 4'
477 │   │     │     │        replace PLACEMENT with ' 4'
478 │   │     │     │        store ' 5' to tPLACE
479 │   │     │     ├──case tPLACE = ' 5'
480 │   │     │     │        replace PLACEMENT with ' 5'
481 │   │     │     │        store ' 6' to tPLACE
482 │   │     │     ├──case tPLACE = ' 6'
483 │   │     │     │        replace PLACEMENT with ' 6'
484 │   │     │     │        store ' 7' to tPLACE
485 │   │     │     ├──case tPLACE = ' 7'
486 │   │     │     │        replace PLACEMENT with ' 7'
487 │   │     │     │        store ' 8' to tPLACE
488 │   │     │     ├──case tPLACE = ' 8'
489 │   │     │     │        replace PLACEMENT with ' 8'
490 │   │     │     │        store ' 9' to tPLACE
491 │   │     │     ├──case tPLACE = ' 9'
492 │   │     │     │        replace PLACEMENT with ' 9'
493 │   │     │     │        store '10' to tPLACE
494 │   │     │     ├──case tPLACE = '10'
495 │   │     │     │        replace PLACEMENT with '10'
496 │   │     │     │        store ' 1' to tPLACE
497 │   │     │     └──endcase
498 │   │     │        skip
499 │   │     └──enddo
500 │   ├──otherwise
501 │   │     ┌──do while TABLE = tTABLE
502 │   │     │     ┌──do case
503 │   │     │     ├──case tPLACE = ' 1'
504 │   │     │     │        replace PLACEMENT with ' 1'
505 │   │     │     │        store ' 2' to tPLACE
506 │   │     │     ├──case tPLACE = ' 2'
507 │   │     │     │        replace PLACEMENT with ' 2'
508 │   │     │     │        store ' 1' to tPLACE
509 │   │     │     └──endcase
510 │   │     │        skip
511 │   │     └──enddo
512 │   └──endcase
513 └──enddo
514
515 <===return
516
517
518     ************************************************
519     *                PROCEDURE SHOP                *
520     *             Shop Laydown Format              *
521     ************************************************
522
523     PROC SHOP
```

```
524
525     store 'SH-00' to tFIND
526     do DIVIDE
527     store '                ' to title4
528     do LINE
529     store tCondit1 to t_cond
530     do GANTT_ln
531     x = x+1
532     store BOLD_1+'    SHOP'+normal_1 to title4
533     do LINE
534     do DATA_ln
535     x = x+1
536     store '                ' to title4
537     store tCondit2 to t_cond
538     do LINE
539     do GANTT_ln
540     x = x+1
541     store '   ASSEMBLIES   ' to title4
542     do LINE
543     do DATA_ln
544     x = x+1
545     store '                ' to title4
546     store tCondit3 to t_cond
547     do LINE
548     do GANTT_ln
549     x = x+1
550     do LINE
551     do DATA_ln
552     x = x+1
553     store tCondit4 to t_cond
554     do LINE
555     do GANTT_ln
556     x = x+1
557     do LINE
558     do DATA_ln
559     x = x+1
560     store tCondit5 to t_cond
561     do LINE
562     do GANTT_ln
563     x = x+1
564     do LINE
565     do DATA_ln
566     x = x+1
567     store tCondit6 to t_cond
568     do LINE
569     do GANTT_ln
570     x = x+1
571     do LINE
572     do DATA_ln
573     x = x+1
574     store tCondit7 to t_cond
575     do LINE
576     do GANTT_ln
577     x = x+1
578     do LINE
579     do DATA_ln
580     x = x+1
581     store tCondit8 to t_cond
582     do LINE
583     do GANTT_ln
584     x = x+1
585     do LINE
586     do DATA_ln
587     x = x+1
588     store tCondit9 to t_cond
589     do LINE
590     do GANTT_ln
591     x = x+1
592     do LINE
593     do DATA_ln
594     x = x+1
595     store tCondit10 to t_cond
596     do LINE
597     do GANTT_ln
598     x = x+1
```

```
599     do LINE
600     do DATA_ln
601     x = x+1
602     do BOTTOM
603     eject
604
605 <===return
606
607
608     ************************************************
609     *             PROCEDURE TABLE1                 *
610     *          Table 1 Laydown Format              *
611     ************************************************
612
613     PROC TABLE1
614
615     *** POSITION 01-01 ***
616     store '01-01' to tFIND
617     * Top Line (Dividing Line) *
618     do DIVIDE
619     * Line 1 (Top Line of Scheduled Work) *
620     store '                  ' to title4
621     do LINE
622     store tCondit1 to t_cond
623     do GANTT_ln
624     x = x+1
625     * Line 2 (Information Line for Gantt Line #1 )
626     store BOLD_1+'  01-01'+normal_1 to title4
627     do LINE
628     store tCondit1 to t_cond
629     do DATA_ln
630     x = x+1
631     * Line 3 (Line #2 of Gantt Schedule    ) *
632     store "   54' x 60'    " to title4
633     do LINE
634     store tCondit2 to t_cond
635     do GANTT_ln
636     x = x+1
637     * Line 4 (Information for Gantt Line #2) *
638     store '                  ' to title4
639     do LINE
640     store tCondit2 to t_cond
641     do DATA_ln
642     x = x+1
643     * Line 5 (Dividing Line) *
644     do DIVIDE
645
646     *** POSITION 01-02 ***.
647     store '01-02' to tFIND
648     * Line 1 (Top Line of Scheduled Work) *
649     store '                  ' to title4
650     do LINE
651     store tCondit1 to t_cond
652     do GANTT_ln
653     x = x+1
654     * Line 2 (Information Line for Gantt Line #1 )
655     store BOLD_1+'  01-02'+normal_1 to title4
656     do LINE
657     store tCondit1 to t_cond
658     do DATA_ln
659     x = x+1
660     * Line 3 (Line #2 of Gantt Schedule    ) *
661     store "   54' x 60'    " to title4
662     do LINE
663     store tCondit2 to t_cond
664     do GANTT_ln
665     x = x+1
666     * Line 4 (Information for Gantt Line #2) *
667     store '                  ' to title4
668     do LINE
669     store tCondit2 to t_cond
670     do DATA_ln
671     x = x+1
672     * Line 5 (Dividing Line) *
673     do DIVIDE
```

```
674
675     *** POSITION 01-03 ***
676     store '01-03' to tFIND
677     • Line 1 (Top Line of Scheduled Work) •
678     store l                  to title4
679     do LINE
680     store tConditl to t_cond
681     do GANTT_ln
682     x = x+1
683     * Line 2 (Information Line for Gantt Line #1 )
684     store BOLD_l+# 01-03'+ normal_l to title4
685     do LINE
686     store tConditl to t-cond
687     do DATA_ln
688     x = X+l
689     * Line 3 (Line #2 of Gantt Schedule ) *
690     store     54' x 60'     11 to title4
691     & LINE
692     store tCondit2 to t-cord
693     do GAHTT-in
694     x = X+l
695     * Line 4 (Information for Gantt Line #2) *
696     store l                  ' to title4
697     do LINE
698     store tCondit2 to t_cond
699     do DATA-in
700     x = X+l
701     • Line 5 (Dividing Line) •
        do DIVIDE
702
704     *** POSITION 01-04 ***
705     store '01-04'to tFIND
706     • Line 1 (Top Line of Scheduled Work) *
707     store                    to title4
708     do LINE
709     store tConditl to t_cond
710     do GANTT-in
711       = X+l
712     * Line 2 (Information Line for Gantt Line #1 )
713     store BOLD-l+'01-04 '+normal-l to title4
714     do LINE
715     store tCmditl to t_cond
716     do DATA_ln
717     x = X+l
718     • Line 3 (Line #2 of Gantt Schedule ) *
719     store "    541 x 60'     ''to title4
720     do LINE
721     store tCondit2 to t_cond
722     do GANTT-in
723     x = X+l
724     * Line 4 (Information for Gantt Line #2) •
725     store 1                  'to title4
726     do LINE
727     store tCondit2 to t_cond
728     do DATA_ln
729     x = X+l
730     • Line 5 (Dividing Line) *
731     do DIVIDE
732
733     ~ POSITION 01-05 • *
734     store '01-05'to tFIND
735     • Line 1 (Top Line of Scheduled Work) •
736     store                    'to title4
737     do LINE
738     store tconditl to t_cond
739     do GANTT-in
740     x = X+l
741     • Line 2 (Information Line for Gantt Line #1 )
742     store BOLD_1+'  01-05'+normal_1 to title4
743     do LINE
744     store tconditl to t-cond
745     do DATA-in
746     x = X+1
747     * Line 3 (Line #2 of Gantt Schedule ) *
748     store     54' x 60'     to title4
```

```
749     do LINE
750     store tCondit2 to t_cond
751     do GANTT-in
752     x = X+l
753     * Line 4 (Information for Gantt Line #2) *
754     store a                 ' to title4
755     do LIME
756     store tCondit2 to t_cond
757     do DATA_ln
758     x = X+1
759     • Line 5 (Dividing Line) *
760     do DIVIDE
761
762     PDSITIDN 01-06 ***
763     store 101-06S to tFINO
764     • Line 1 (Top Line of Scheduled Work) *
765     store '                 ' to title4
766     do LINE
767     store tConditl to t-cond
768     do GANTT-in
769     x x X+l
770     * Line 2 (Information Line for Gantt Line #1 )
771     store BOLO-I+$01-06 '+normal I to title4
772     do LINE
773     store tconditl to t-cond
774     do DATA_ln
775     x = X+1
776     • Line 3 (Line #2 of Gantt Schedule ) *
777     store II    54' X 60)I     "  to title4
778     do LINE
779     store tCondit2 to t_cod
780     do GANTT_ln
781     x = X+l
782     • Line 4 (Information for Gantt Line #2) *
783     store I                 ' to title4
784     do LINE
785     store tCondit2 to t-cond
786     do DATA-in
787     x = x(+1
788     • Line 5 (Dividing Line) •
789     do DIVIDE
790
791     *** POSITION 01-07 ***
792     store '01-07' to tFIND
793     • Line 1 (Top Line of Scheduled work) •
794     store I                 ' to title4
795     do LINE
796     store tConditl to t-cond
797     do GANTT_ln
798     x = X+1
799     * Line 2 (Information Line for Gantt Line #l )
800     store BOLD_I+' 01-07 normal_l to title4
801     do LINE
802     store tConditl to t-cod
803     do DATA_ln
804     x = x+1
805     • Line 3 (Line #2 of Gantt Schedule ) •
806     store "    54' x 60,    II to title4
807     do LINE
808     store tCondit2 to t_cond
809     do GANTT in
810     x = X+1
811     * Line 4 (Information for Gantt Line #2) *
812     store                   to title4
813     do LINE
814     store tcondit2 to t_cond
815     do DATA-in
816     x xX+1
817     • Line 5 (Dividing Line) *
818     do DIVIDE
819
820     *** POSITION 01-08 ***
821     store 101-081 to tFIND
822     • Line 1 (Top Line of Scheduled Work) -
823     store '                 ' to title4
```

```
824     do LINE
825     store tConditl to t_cond
826     do GANTT_ln
827     x = X+l
828     • Line 2 (Information Line for Gantt Line #1 )
829     store SOLD-1+1 01-08'+normal_l to title4
830     do LINE
831     store tConditl to t-cord
832     do DATA_ln
833     x = X+l
834     * Line 3 (Line #2 of Gantt Schedule ) *
835     store "   54' x 60'     " to title4
836     do LINE
837     store tCondit2 to t_cond
838     do GANTT_ln
839     x = X+l
840     • Line 4 (Information for Gantt Line #2) *
841     store ¹                ¹to title4
842     do LINE
243     store tCondit2 to t_cond
844     do DATA_ln
845     x = X+l
846     * Line 5 (Dividing Line) •
847     do DIVIDE
848
849     *** POSITION 01-09 ***
850     store '01-09' to tFIND
851     * Line 1 (Top Line of Scheduled Work) *
852     store ⁴                ¹to title4
853     do LINE
854     store tConditl to t_cmd
855     do GANTT-in
856     x = x+1
857     • Line 2 (Information Line for Gantt Line #1 )
858     store SOLD_l+'01-09'+normal_l to title4
859     do LINE
860     store tConditl to t-cond
861     do DATA-in
862     x = X+l
863     * Line 3 (Line #2 of Gantt Schedule ) •
864     store "   54' x 60',     " to title4
865     do LINE
866     store tcondit2 to t_cond
867     do GANTT_ln
868     x = X+l
869     • Line 4 (Information for Gantt Line #2) -
870     store                  to title4
871     do LINE
872     store tcondit2 to t-cond
873     do DATA_ln
874     x = X+l
875     * Line 5 (Dividing Line) •
876     do DIVIDE
877
878     * POSITION 01-10 • -
879     store O1-10' to tFIND
880     * Line 1 (Top Line of Scheduled Work) *
881     store '                to title4
882     do LINE
883     store tconditl to t-cond
884     do GANTT_ln
885     x = X+1
886     * Line 2 (Information Line for Gantt Line #1 )
887     store SOLD_l+'O1-lO'+normal 1 to title4
888     do LINE
889     store tconditl to t_cond
890     do DATA_ln
891     x = X+l
892     * Line 3 (Line #2 of Gantt Schedule ) •
893     store ᴺ   54' x 60'      to title4
894     do LINE
895     store tCondit2 to t-cond
896     do GANTT_ln
897     x = X+l
898     * Line 4 (Information for Gantt Line #2) *
```

```
899     store '                ' to title4
900     do LINE
901     store tCondit2 to t_cond
902     do DATA_ln
903     x = x+1
904     * Line 5 (Bottom Line) *
905     do BOTTOM
906     eject
907
908 <===return
909
910
911
912     ***********************************************
913     *             PROCEDURE TABLE2                *
914     *          Table 2 Laydown Format             *
915     ***********************************************
916
917     PROC TABLE2
918
919     *** POSITION 02-01 ***
920     store '02-01' to tFIND
921     * Top Line (Dividing Line) *
922     do DIVIDE
923     * Line 1 (Top Line of Scheduled Work) *
924     store '                ' to title4
925     do LINE
926     store tCondit1 to t_cond
927     do GANTT_ln
928     x = x+1
929     * Line 2 (Information Line for Gantt Line #1 )
930     store BOLD_1+'  02-01'+normal_1 to title4
931     do LINE
932     store tCondit1 to t_cond
933     do DATA_ln
934     x = x+1
935     * Line 3 (Line #2 of Gantt Schedule    ) *
936     store "   40' x 50'    " to title4
937     do LINE
938     store tCondit2 to t_cond
939     do GANTT_ln
940     x = x+1
941     * Line 4 (Information for Gantt Line #2) *
942     store '                ' to title4
943     do LINE
944     store tCondit2 to t_cond
945     do DATA_ln
946     x = x+1
947     * Line 5 (Dividing Line) *
948     do DIVIDE
949
950     *** POSITION 02-02 ***
951     store '02-02' to tFIND
952     * Line 1 (Top Line of Scheduled Work) *
953     store                  to title4
954     do LINE
955     store tConditl to t-cond
956     do GANTT-in
957     x = X+l
958     * Line 2 (Information Line for Gantt Line #1 )
959     store BOLD-1+1 02-02 '+normal_l to title4
960     do LINE
961     store tCondit1 to t-cond
962     do DATA_ln
963     x = X+l
964     * Line 3 (Line #2 of Gantt Schdule ) *
965     store " 40' x 50'    " to title4
966     do LINE
967     store tCondit2 to t-cond
968     do GANTT-in
969     x = X+l
970     * Line 4 (Information for Gantt Line #2) *
971     store '                ' to title4
972     do LINE
973     store tCondit2 to t-cond
```

```
974     do DATA-in
975     x = X+1
976     • Line 5 (Dividing Line) •
977     do DIVIDE
978
979     * POSITION 02-03 ***
980     store '02-03' to tFIND
981     • Line 1 (Top Line of Scheduld Work) •
982     store ';                    to title4
983     do LINE
984     store tConditl to t_cond
985     do GANTT_ln
986     x = X+l
987     • Line 2 (Information Line for Gantt Line #l )
988     store BOLD_l+l 02-03' +normal_l to title4
989     do LINE
990     store tConditl to t_cond
991     do DATA-tn
992     x = X+l
993     • Line 3 (Line #2 of Gantt Schedule ) •
994     store "   40' x 50'     " to title4
995     do LINE
996     store tCondit2 to t-cond
997     do GANTT_ln
998     x = X+l
999     • Line 4 (Information for Gantt Line #2) •
1000    store '                   ' to title4
1001    do LINE
1002    store tCondit2 to t-cond
1003    do DATA-in
1004    x = X+l
1005    * Line 5 (Dividing Line) •
1006    do DIVIDE
1007
1008    *** POSITION 02-04 ***
1009    store '02-04' to tFIND
1010    • Line 1 (Top Line of Scheduled Work) •
1011    store                    to title4
1012    do LINE
1013    store tConditl to t_cond
1014    do GANTT-in
1015    x = X+l
1016    • Line 2 (Information Line for Gantt Line #1 )
1017    store BOLD-l+'02-04 '+normal l to title4
1018    do LINE
1019    store tConditl to t_cond
1020    do DATA-in
1021    x = X+l
1022    • Line 3 (Line #2 of Gantt Schedule ) *
1023    store "   40'x50'     " to title4
1024    do LINE
1025    store tCondit2 to t_cond
1026    do GANTT-in
1027    x = X+l
1028    * Line 4 (Information for Gantt Line #2) *
1029    store '                  'to title4
1030    do LINE
1031    store tCondit2 to t_cond
1032    do DATA-in
1033    x = X+l
1034    • Line 5 (Dividing Line) *
1035    do DIVIDE
1036
1037
1038    *** POSITION 02-05 • *
1039    store '02-05' to tFIND
1040    * Line 1 (Top Line of Scheduled Work) *
1041    store '                   ' to title4
1042    do LINE
1043    store tCoditl to t-cond
1044    do GANTT_ln
1045    x = X+l
1046    • Line 2 (Information Line for Gantt Line #1 )
1047    store BOLD-1+' 02-05'+normal l to title4
1048    do LINE
```

```
1049      store tConditl to t-cond
1050      do DATA-in
1051      x = X+l
1052      * Line 3 (Line #2 of Gantt Schedule ) •
1053      store "    40' x 50'     "  to title4
1054      do LINE
1055      store tCondit2 to t-cond
1056      do GANTT-in
1057    x  =  X+l
1058      * Line 4 (Information for Gantt Line #2) •
1059      store                    to title4
1060      do LINE
1061      store tcondit2 to t_cond
1062      do DATA-in
1063    x  =  X+l
1064      * Line 5 (Bottom Line) *
1065      do BOTTOM
1066      eject
1067
1068  <===return
1069
1070
1071      ********************************************
1072      *            PROCEDURE AREA_A              *
1073      *          Area A Laydown Format           *
1074      ********************************************
1075
1076      PROC AREA_A
1077
1078      *** POSITION A-1 ***
1079      store  A-01 to tFIND
1080      do DIVIDE
1081      store                    to title4
1082      do LINE
1083      store tconditl to t_cond
1084      do GANTT-in
1085      x = X+l
1086      store BOLD-l+'    A-1 '+normal_l to title4
1087      do LIME
1088      store tconditl to t-cond
1089      do DATA-in
1090      x = x+1
1091      store     40 x 60'     "  to title4
1092      do LINE
1093      store tCondit2 to t_cond
1094      do GANTT-in
1095      x = X+l
1096      store '                  to title4
1097      do LINE
1098      store tcondit2 to t-cond
1099      do DATA-in
1100      x = X+l
1101      do DIVIDE
1102
1103      *** POSITISIN A-2 ***
1104      store 'A-02' to tFIND
1105      store '                  '  to title4
1106      do LINE
1107      store tconditl to t_cond
1108      do GANTT-in
1109      x = x+1
1110      store BOLD-1+:    A-2 '+normal 1 to title4
1111      do LINE
1112      store tconditl to t_cond
1113      do DATA-in
1114      x = X+l
1115      store     40' x 60'      to title4
1116      do LINE
1117      store tcondit2 to t-cond
1118      do GANTT_ln
1119      x = X+l
1120      store                    '  to title4
1121      do LIME
1122      store tCondit2 to t-cond
1123      do DATA-in
```

```
1124    x = X+l
1125    do DIVIDE
1126
1127    *** POSITION A-3 • *
1128    store ' A-03' to tFIND
1129    store '                ' to title4
1130    do LINE
1131    store tConditl to t_cond
1132    do GANTT-in
1133    x = X+l
1134    store BOLD_l+'   A.3 l+normal l to title4
1135    do LINE
1136    store tcondit1 to t-cond
1137    do DATA-in
1138    x = X+l
1139    store "   40' x 60'    " to title4
1140    do LINE
1141    store tCondit2 to t_cond
1142    do GANTT_ln
1143    x = X+l
1144    store                  to title4
1145    do LINE
1146    store tCondit2 to t_cond
1147    do DATA_ln
1148    x = X+l
1149    do DIVIDE
1150
1151    *** POSITION A-4 ***
1152    store  A-04' to tFIND
1153    store '                ' to title4
1154    do LINE
1155    store tConditl to t_cond
1156    do GANTT-in
1157    x = X+l
1158    store BOLD_l+'   A.4 l+normal to title4
1159    do LINE
1160    store tConditl to t_cond
1161    do DATA-in
1162    x = X+l
1163    store "   4o' x 60'    " to title4
1164    do LINE
1165    store tCondit2 to t_cond
1166    do GANTT-in
1167    x = X+l
1168    store '                ' to title4
1169    do LINE
1170    store tCondit2 to t_cond
1171    do DATA_ln
1172      X= X+l
1173    do DIVIDE
1174
1175    *** POSITION A-5 ***
1176    store 'A-05' to tFIND
1177    store                 'to title4
1178    do LINE
1179    store tConditl to t_cond
1180    do GANTT-in
1181    x = X+l
1182    store BOLD_l+'   A-5 l+normal to title4
1183    do LINE
1184    store tConditl to t_cond
1185    do DATA-in
1186    x = X+l
1187    store'   40' x 60'    " to title4
1188    do LINE
1189    store tcondit2 to t-cond
1190    do GANTT_ln
1191    x = X+l
1192    store                  to title4
1193    do LINE
1194    store tCondit2 to t_cond
1195    do DATA-in
1196    x = X+l
1197    do DIVIDE
1198
```

```
1199      *** POSITION A-6 ***
1200      store ' A-(06' to tFIND
1201      store '                ' to title4
1202      do LINE
1203      store tconditl to t-cond
1204      do GANTT_ln
1205      x = X+l
1206      store BOLD-1+'    A-6 l+normal to title4
1207      do LINE
1208      store tCondit 1 to t_cond
1209      do DATA_ln
1210      x = X+1
1211      store     40' x60'      to title4
1212      do LINE
1213      store tCondit2 to t_cond
1214      do GANTT-in
1215      x = X+l
1216      store                   to title4
1217      do LINE
1218      store tCondit2 to t-cond
1219      do DATA-in
1220      x = x+1
1221      do DIVIDE
1222
1223      *** POSITION A-7 ***
1224      store   A-07' to tFIND
1225      store #                 'to title4
1226      do LINE
1227      store tconditl to t_cond
1228      do GANTT_ln
1229      x = X+l
1230      store BOLD-l+'    A.7 l+normal 1 to title4
1231      do LINE
1232      store tConditl to t-cond
1233      do DATA-in
1234      x = X+l
1235      store     30' X60'    to title4
1236      do LINE
1237      store tCondit2 to t_cond
1238      do GANTT_ln
1239      x = x+1
1240      store                   to title4
1241      do LINE
1242      store tCondit2 to t_cond
1243      do DATA-in
1244      x = X+l
1245      do DIVIDE
1246
1247      *** POSITION A-8 ***
1248      store ' A-08' to tFIND
1249      store                   to title4
1250      do LINE
1251      store tconditl to t-cond
1252      do GANTT_ln
1253      X=X+1
1254      store BOLD-1+'    A-8  '+normal  to  title4
1255      do LINE
1256      store tconditl to t-cond
1257      do DATA-in
1258      x = X+1
1259      store     40' x 60'     "  to title4
1260      do LINE
1261      store tCondit2 to t-cond
1262      do GANTT-in
1263      x = X+1
1264      store                   to title4
1265      do LINE
1266      store tCondit2 to t_cond
1267      do DATA-in
1268      x = X+l
1269      do BOTTOM
1270      eject
1271
1272  <=..return
1273
```

```
1274
1275    ***********************************************
1276    •              PROCEDURE AREA_B                 *
1277    •           Area B Laydown Format               •
1278    ***********************************************
1279
1280    PROC AREA_B
1281
1282    *** POSITION B-1 • *
1283    store B-01 'to tFIND
1284    do DIVIDE
1285    store                    to title4
1286    do LINE
1287    store tConditl to t-cond
1288    do GANTT_ln
1289    x = X+l
1290    store BOLD_l+     B-1 '+normal_l to title4
1291    do LINE
1292    store tConditl to t_cond
1293    do DATA_ln
1294    x = X+l
1295    store     40' x 60'      to title4
1296    do LINE
1297    store tCmdit2 to t_cond
1298    do GANTT-in
1299    x = X+l
1300    store '                  to title4
1301    do LINE
1302    store tCondit2 to t-cond
1303    do DATA-in
1304    x = X+l
1305    do DIVIDE
1306
1307    *** POSITION B-2 ***
1308    store E-02' to tFIND
1309    store                    to title4
1310    do LINE
1311    store tconditl to t_cond
1312    do GANTT-in
1313    x = X+l
1314    store BOLD-l+'   B-2 l+normal to title4
1315    do LINE
1316    store tconditl to t_cond
1317    do DATA_ln
1318    x = X+1
1319    store     40' x 60'      to title4
1320    do LINE
1321    store tCondit2 to t_cond
1322    do GANTT_ln
1323    x = X+l
1324    store                    to title4
1325    do LINE
1326    store tCondit2 to t_cond
1327    do DATA_ln
1328    x = X+l
1329    do DIVIDE
1330
1331    *** POSITION B-3
1332    store B-03 to tFIND
1333    store                    to title4
1334    do LINE
1335    store tconditl to t_cond
1336    do GANTT-in
1337    x = X+l
1338    store BOLD_l+'   B-3 l+normal l to title4
1339    do LINE
1340    store tconditl to t_cond
1341    do DATA_ln
1342    x = X+l
1343    store     40' x 60'    to title4
1344    do LINE
1345    store tCondit2 to t-cond
1346    do GANTT_ln
1347    x = X+1
1348    store                    to title4
```

```
1349    do LIME
1350    store tCondit2 to t-cond
1351    do DATA_ln
1352    x = X+1
1353    do DIVIDE
1354
1355    *** POSITION B-4 ***
1356    store  B-04' to tFIND
1357    store                    to title4
1358    do LIME
1359    store tConditl to t-cond
1360    do GANTT-in
1361    x = X+l
1362    store BOLD-l+'  B.4 l+normal l to title4
1363    do LINE
1364    store tConditl to t-cond
1365    do DATA-in
1366    x = X+1
1367    store     40' x 60'     to tittle4
1368    do LINE
1369    store tCondit2 to t-cond
1370    do GANTT_in
1371    x = X+l
1372    store                    to title4
1373    do LINE
1374    store tcondit2 to t_cond
1375    do DATA_ln
1376    x = X+l
1377    do DIVIDE
1378
1379    *** POSITION B-5 ***
1380    store  B-05' to tFIND
1381    store                    to title4
1382    do LINE
1383    store tconditl to t_cond
1384    do GANTT-in
1385    x = X+l
1386    store BOLD-l+  B-5 l+normal 1 to title4
1387    do LINE
1388    store tconditl to t-cond
1389    do DATA-in
1390    x = X+l
1391    store     40' x 60'     "  to title4
1392    do LINE
1393    store tCondit2 to t_cond
1394    do GANTT.in
1395    x = X+1
1396    store                    to title4
1397    do LINE
1398    store tCondit2 to t_cond
1399    do DATA-in
1400    x = X+l
1401    do DIVIDE
1402
1403    *** POSITION B-6 • -
1404    store  B-06' to tFIND
1405    store                    to title4
1406    do LINE
1407    store tconditl to t_cond
1408    do GANTT-in
1409    x = X+l
1410    store BOLD-1+'  B-6 l+normal 1 to title4
1411    do LINE
1412    store tCoditl to t_cond
1413    do DATA_ln
1414    x = X+1
1415    store     40' x 60'     "  to title4
1416    do LINE
1417    store tCondit2 to t_cond
1418    do GANTT_ln
1419    x = X+l
1420    store                    1  to title4
1421    do LINE
1422    store tCondit2 to t_cond
1423    do DATA-in
```

```
1424      x = X+1
1425      do DIVIDE
1426
1427      *** POSITION B-7 ***
1428      store ' B-07' to tFIND
1429      store '                  I to title4
1430      do LINE
1431      store tConditl to t-cond
1432      do GANTT-in
1433      x = X+I
1434      store BOLD_I+'    B-7 I+normal I to title4
1435      do LINE
1436      store tCondit1 to t_cond
1437      do DATA-in
1438      x = X+1
1439      store     4'0, x 60'     1  to  title4
1440      do LINE
1441      store tCondit2 to t_cond
1442      do GANTT-in
1443      x = X+1
1444      store '                  I to title4
1445      do LINE
1446      store tCondit2 to t-cond
1447      do DATA_In
1448      x = X+1
1449      do DIVIDE
1450
1451
1452      ***  POSITION  B-8  *
1453      store'  6-08'to tFIND
1454      store '                  '  to  title4
1455      do LINE
1456      store tconditl to t_cond
1457      do GANTT_In
1458      x = X+I
1459      store BOLD-I +'    B.8 '+normal 1 to title4
1460      do LINE
1461      store tConditl to t_cond
1462      do DATA-in
1463      x = X+I
1464      store "    40' x 60'     "  to  title4
1465      do LINE
1466      store tCondit2 to t_cond
1467      do GANTT_In
1468      x = X+I
1469      store '                  '  to  title4
1470      do LINE
1471      store tCondit2 to t_cond
1472      do DATA-in
1473      x = X+I
1474      doBOTTOM
1475      eject
1476 <===return
1477
1478
1479      ***********************************************
1480      *             PROCEDURE BOTTOM                *
1481      *       Prints Bottom Line of Format          *
1482      ***********************************************
1483
1484      PROC BOTTOM
1485
1486      @ x,y     say chr(200)
1487      @ x,y+1   say replicate(chr(205),20)
1488      @ x,y+21  say chr(202)
1489      @ x,y+22  say replicate(chr(205),8)
1490      @ x,y+30  say ' X = M/V WELL PLANNED '
1491      @ x,y+72  say replicate(chr(205),132)
1492      @ x,y+204 say chr(188)
1493      x = x+1
1494
1495      do LEGEND
1496
1497 <===return
1498
```

```
1499
1500
1501     ************************************************
1502     *              PROCEDURE GANTT ln              *
1503     • Prints a Gantt Bar Line in a Laydoun Format •
1504 ************************************************************************
1505
1506     PROC GANTT in
1507
1508     go top
1509     store ' ' to tGO
1510     SEEK tFIND
1511 ┌──if .not. eof()
1512 │   ┌──do while TABLE = tFIND
1513 │   │   ┌──if LFDATE > tSTART .and. LSDATE < tCOMPLETE
1514 │   │   │       store 'Y' to tGO
1515 │   <===<===<===exit
1516 │   │   ├──else
1517 │   │   │   ┌──if LSDATE > tCOMPLETE
1518 │   <===<===<===<===exit
1519 │   │   │   └──endif
1520 │   │   │      skip
1521 │   │   └──endif
1522 │   └──enddo
1523 │   ┌──if tGO = 'Y'
1524 │   │   ┌──do while LFDATE > tSTART .and. LSDATE < tCOMPLETE .and. TABLE = tFIND
1525 │   │   │   ┌──if LSDATE <= LFDATE
1526 │   │   │   │   ┌──if &t_cond
1527 │   │   │   │   │   ┌──do case
1528 │   │   │   │   │   ├──case substr(ID,2,2) = '71'
1529 │   │   │   │   │   │       store chr(219) to tGANTT
1530 │   │   │   │   │   ├──case substr(ID,2,4) = '7251'
1531 │   │   │   │   │   │       store chr(202) to tGANTT
1532 │   │   │   │   │   ├──case substr(ID,2,4) = '7252'
1533 │   │   │   │   │   │       store chr(203) to tGANTT
1534 │   │   │   │   │   ├──case substr(ID,2,4) = '7254'
1535 │   │   │   │   │   │       store chr(178) to tGANTT
1536 │   │   │   │   │   └──endcase
1537 │   │   │   │   │   ┌──if LSDATE > tSTART
1538 │   │   │   │   │   │   ┌──if LFDATE < tCOMPLETE
1539 │   │   │   │   │   │   │       @ x,y+21+(LSDATE-tSTART) say replicate(tGANTT,LFDATE-LSDATE+1)
1540 │   │   │   │   │   │   │   ┌──if tCOMPLETE-LFDATE < 6
1541 │   │   │   │   │   │   │   │       @ x,y+22+(183) say SIZE
1542 │   │   │   │   │   │   │   ├──else
1543 │   │   │   │   │   │   │   │       @ x,y+22+(LSDATE-tSTART)+(LFDATE-LSDATE) say SIZE
1544 │   │   │   │   │   │   │   └──endif
1545 │   │   │   │   │   │   ├──else
1546 │   │   │   │   │   │   │       @ x,y+21+(LSDATE-tSTART) say replicate(tGANTT,tCOMPLETE-LSDATE)
1547 │   │   │   │   │   │   │       @ x,y+22+(183) say SIZE
1548 │   │   │   │   │   │   └──endif
1549 │   │   │   │   │   ├──else
1550 │   │   │   │   │   │       @ x,y+22 say replicate(tGANTT,LFDATE-tSTART)
1551 │   │   │   │   │   │       @ x,y+22+(LFDATE-tSTART) say SIZE
1552 │   │   │   │   │   └──endif
1553 │   │   │   │   └──endif
1554 │   │   │   └──endif
1555 │   │   │      skip
1556 │   │   └──enddo
1557 │   └──endif
1558 └──endif
1559
1560
1561 <===return
1562
1563
1564
1565     ************************************************************
1566     *                   PROCEDURE DATA_ln                      *
1567     *  Prints a Data Information Line in a Laydown Format      *
1568     ************************************************************
1569     ****************************
1570
1571     PROC DATA_ln
1572
1573     go top
```

```
1574      store ' ' to tGO
1575      SEEK tFIND
1576    ┌─if .not. eof()
1577    │   ┌─do while TABLE = tFIND
1578    │   │   ┌─if LFDATE > tSTART .and. LSDATE < tCOMPLETE
1579    │   │   │     store 'Y' to tGO
1580    │   <===<===<===exit
1581    │   │   ├─else
1582    │   │   │     skip
1583    │   │   └─endif
1584    │   └─enddo
1585    │   ┌─if tGO = 'Y'
1586    │   │   ┌─do while LFDATE > tSTART .and. LSDATE < tCOMPLETE .and. TABLE = tFIND
1587    │   │   │   ┌─if LSDATE <= LFDATE
1588    │   │   │   │   ┌─if &t_cond
1589    │   │   │   │   │   ┌─if LSDATE > tSTART
1590    │   │   │   │   │   │   ┌─if LFDATE < tCOMPLETE
1591    │   │   │   │   │   │   │   ┌─if substr(ID,6,3) < '500'
1592    │   │   │   │   │   │   │   │     @ x,y+21+(LSDATE-tSTART) say normal_2+substr(ID,1,1)+' A2'+'-'+substr(ID,6,3)+normal_1
1593    │   │   │   │   │   │   │   ├─else
1594    │   │   │   │   │   │   │   │     @ x,y+21+(LSDATE-tSTART) say normal_2+substr(ID,1,1)+' A1'+'-'+substr(ID,6,3)+normal_1
1595    │   │   │   │   │   │   │   └─endif
1596    │   │   │   │   │   │   │     @ x,y+21+(LSDATE-tSTART) say ' '
1597    │   │   │   │   │   │   ├─else
1598    │   │   │   │   │   │   │   ┌─if tCOMPLETE-LSDATE > 8
1599    │   │   │   │   │   │   │   │   ┌─if substr(ID,6,3) < '500'
1600    │   │   │   │   │   │   │   │   │     @ x,y+21+(LSDATE-tSTART) say normal_2+substr(ID,1,1)+' A2'+'-'+substr(ID,6,3)+norm
1601    │   │   │   │   │   │   │   │   ├─else
1602    │   │   │   │   │   │   │   │   │     @ x,y+21+(LSDATE-tSTART) say normal_2+substr(ID,1,1)+' A1'+'-'+substr(ID,6,3)+norm
1603    │   │   │   │   │   │   │   │   └─endif
1604    │   │   │   │   │   │   │   │     @ x,y+21+(LSDATE-tSTART) say ' '
1605    │   │   │   │   │   │   │   └─endif
1606    │   │   │   │   │   │   └─endif
1607    │   │   │   │   │   ├─else
1608    │   │   │   │   │   │   ┌─if substr(ID,6,3) < '500'
1609    │   │   │   │   │   │   │     @ x,y+22 say normal_2+substr(ID,1,1)+' A2'+'-'+substr(ID,6,3)+normal_1
1610    │   │   │   │   │   │   ├─else
1611    │   │   │   │   │   │   │     @ x,y+22 say normal_2+substr(ID,1,1)+' A1'+'-'+substr(ID,6,3)+normal_1
1612    │   │   │   │   │   │   └─endif
1613    │   │   │   │   │   │     @ x,y+21+(LSDATE-tSTART) say ' '
1614    │   │   │   │   │   └─endif
1615    │   │   │   │   └─endif
1616    │   │   │   └─endif
1617    │   │   │     skip
1618    │   │   └─enddo
1619    │   └─endif
1620    └─endif
1621
1622
1623  <===return
1624
1625
1626
1627      ************************************************************
1628      *                    PROCEDURE LINE                        *
1629      *  Prints a Background Shade Line in a Laydown Format      *
1630      ************************************************************
1631
1632      PROC LINE
1633
1634      @ x,y     say chr(186)
1635      @ x,y+21  say chr(186)
1636      @ x,y+22  say lineE
1637      @ x,y+204 say chr(186)
1638      @ x,y+2   say title4
1639      @ x,y     say ' '
1640
1641
1642  <===return
1643
1644
1645
1646      ************************************************************
1647      *                    PROCEDURE DIVIDE                      *
1648      *      Prints a Dividing Line in a Laydown Format        *
```

```
1649    ***********************************************
1650
1651      PROC DIVIDE
1652
1653      a x,Y       say chr(204)
1654      a x,Y+I     say replicate(chr(205).20)
1655      a x,Y+21  say chi(206)
1656      a   x,Y+22    say    replicate(chr(205),182)
1657      a x,Y+22 say LineE
1658      a x, Y+204 say chr(185)
1659   X  =  x+1
1660
1661 <===return
1662
1663
1644
1665      ********************************************************
1666      *                  PROCEDURE LEGEND                    *
1667      *      Prints the legend     in a Laydown Format      *
1668      ********************************************************
1669
1670      PROC LEGEND
1671
1672      store chr(219) to tASSY
1673      store chr(202) to tOF_INV
1674      store chr(203) to tOF_UPR
1675      store chr(178) to tOF_POST
1676      a x,y+8   say 'LEGEND:'
1677      x = x + 1
1678      a x,y+10  say 'Assembly: ' + replicate(tASSY,5)+';'
1679      a x,y+27  say 'O/F Pre-Blast Inverted: ' + replicate(tOF_INV,5)+';'
1680      a x,y+58  say 'O/F Pre-Blast UpRight: ' + replicate(tOF_UPR,5)+';'
1681      a x,y+88  say 'O/F Post Blast: ' + replicate(tOF_POST,5)+';'
1682
1683
1984 <===return
1685
1686
1687
1688      ● Formatted by: dANALYST Ver. 7.3a on March 24, 1992 at 9:50 AM.
dANALYST found O error(s), O warning(s), 1688 lines.


All files processed.

dANALYST found O total error(s), O total Warning, 5354 total lines.
```

APPENDIX VI

APPLICATION OF PC-BASED
FACILITIES SIMULATION

Application of PC-Based Facilities Simulation

This report has described the development of a PC-based model which serves as a tool to assist ship yards in developing, updating, and revising schedules, manning, and facility utilization reports. To this point, the report assumed schedules are to be developed based upon a yard's current facility constraints. A shipyard conducting long range planning may wish to look at strategies and schedules based upon a modification to or a different usage of a yard's facilities. What is the best tool to perform this analysis? A team was formed to investigate the applicability of using a stochastic simulation software package to perform this task. This appendix discusses the insights gained by that group.

The first task the group undertook was to address the questions shown below.

- Is the high level manufacturing process that we are focusing on appropriate for stochastic simulation and, if simulated, are we likely to learn anything that we do not already know?
- If simulating the manufacturing process at this level is not expected to yield useful information, are there other opportunities to take advantage of simulation's strengths that would add value to the planning process within a shipyard?

These questions are answered relative to the capabilities of the Integrated Production Planning System (IPPS). The strengths of stochastic simulation lies in its ability to:

- Assign finite resources and reschedule accordingly.
- Utilize a stochastic analysis methodology to account for variations in actual performance.
- System is table driven and, therefore, is well suited to conducting a high number of iterations, what-ifs, and sensitivity studies with various facility scenarios.

- Capability to allow observation of the dynamic performance of a simulated manufacturing process.

In contrast, the IPPS provides the capability to develop and utilize a deterministic model. Although finite resources are assigned and reported against, the system is not used to automatically adjust schedules based on that information.

The output of any computer system, whether it be stochastic simulation or deterministic modeling is dependent upon the input given to the system. A stochastic simulation program requires as its input data regarding statistical distribution of activity durations, statistical distributions of performance criteria (ie. machine downtime) , and a set of prioritization rules that take into account all the factors to be considered in schedule development.

There is a high degree of uncertainty associated with long range planning analysis. When analyzing facilities and work flow strategies that do not currently exist, no historical data is available. Performance of individual pieces of equipment at other locations may be investigated but this may not be completely applicable to the system arrangement you are analyzing. With regards to long range planning, broad assumptions regarding statistical distributions of activity durations and performance criteria that are required for stochastic simulation model may be no more meaningful that the more generalized assumptions required by a deterministic model. Greater detail in analysis probably would not have caused a change in the recommendation with respect to facilities modifications.

It is important to understand the relative benefits of finite vs. infinite capacity systems. The finite capacity model manages a high number of complex resource constraints much more efficiently than one could manage with an infinite capacity system. The downside is that the model becomes very complex when trying to establish a realistic prioritization logic. Prioritization rules

and logic are not always known beforehand. Sequencing and constraints are often varied as assumptions are changed in the process of schedule development. The complexity is again increased when manpower constraints are variable. For example, additional shifts or overtime decisions (which decrease activity duration) can be made based upon task priority. Output is generated indicating resource utilization relative to forward and backward pass. Prioritization is left to the discretion of the planner. The planner manually adjusts the model assumptions as the schedule is refined. After a number of iterations are performed to effectively level the manufacturing process, system output will reflect resource utilization projections as well as a working schedule. However, capacity/schedule analysis becomes quite complex and time consuming as more and more resource constraints are added.

Based upon the insights gained by the group, the following recommendation is made in regards to the use of stochastic simulation for long range planning.

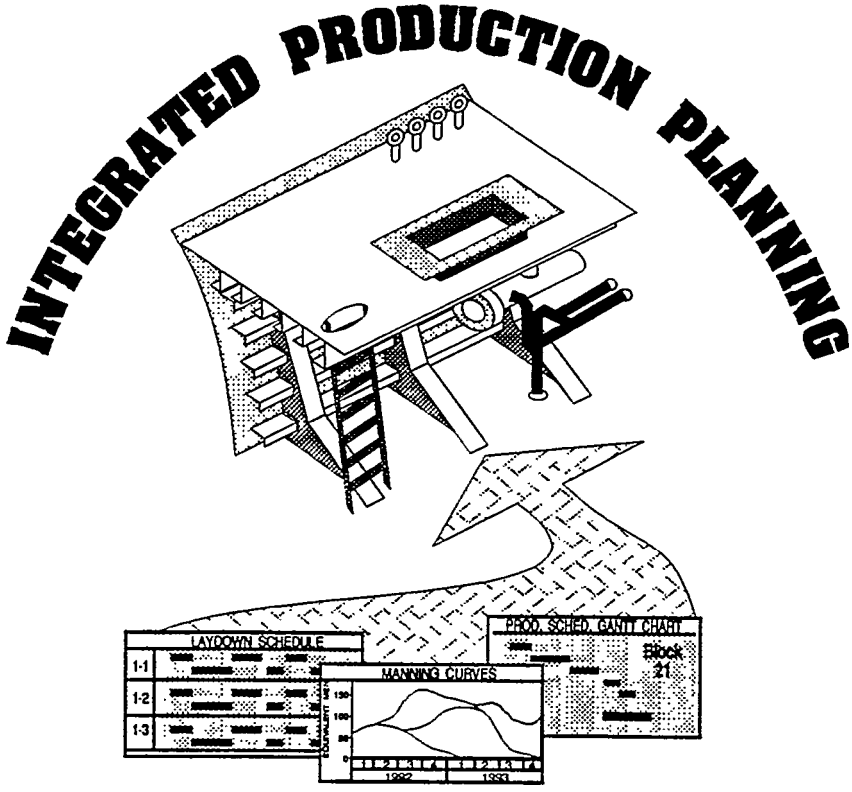
> The dubious benefits associated with the increased detail of a simulation analysis along with the difficulty of defining a complete set of prioritization rules and logic for finite capacity analysis are not worth the increased training, development, and maintenance effort required. Given the data available and broad assumptions that must be made in a long range planning analysis, equally useful information may be gathered through the use of a more simple, deterministic model.

This does not mean, however, that stochastic simulation has no use in a shipyard environment. If a detailed schedule is being developed for a well defined process (ie. a panel line or a machine shop) stochastic simulation may prove a powerful tool. At this level, with established equipment for which historical data ia available, required input of activity durations and down times with their statistical distributions is available. Prioritization rules and logic are clearer at a more detailed level. The rules and logic may be more easily incorporated into the software system so that the software can help produce more meaningful results.

Performing stochastic simulation is an excellent method for resource, capacity, and throughput optimization studies. Simulation can serve as a powerful tool for shipyards. However, like all tools, it is most effective when used properly.

SYSTEM'S

DEMONSTRATION DISK



SYSTEM

SYSTEM'S DEMONSTRATION **DISK**

The "System's Demonstration Disk" is an on screen slide show which gives a graphical overview of the Integrated production Planning System (IPPS). The demonstration disk will step the viewer through both the Open Plan screens used by the IPPS and the screens created by the dBase programs included within this project. To view the demonstration disk, insert the disk into the “B” drive of your computer and type B:\DEMO. Note, the disk requires a VGA monitor to operate.

OF THIS REPORT!

PLEASE RETURN A RESPONSE CARD AFTER READING REPORT.

NSRP READER RESPONSE CARD

We would appreciate your comments on this report. Please take a few minutes to complete and return this postage-paid card. Thank you.

Name _____

Organization _____

Phone _____

Ž Overall Quality of *Report*

☐ Excellent ❑ Good El Fair ☐ Poor

• *Usefulness to You/four Organization*

❑ **Very Useful** ☐ ! Moderately Useful ❑ N/A

Ž *Did/Will your organization implement the results of this project?* ❑ *Yes* ❑ *No*

If not, why? _____

• *How Did You Receive Report?*

❑ Mailed directly to you

El Referred to you by someone else

• *Did/Will You Pass Report On To Someone Else?*

❑ *Yes* ☐ No

• *In Your Opinion, Is Anything Missing That Would Make This Report Better?*

☐ Yes _____

_____

• *Genera/ Comments*

_____

_____

NSRP 024

NSRP READER RESPONSE CARD

We would appreciate your comments on this report. Please take a few minutes to complete and return this postage-paid card. Thank you.

Name _____

Organization _____

Phone _____

• *Overall Quality of Report*

❑ Excellent ☐ Good ☐ Fair •I Poor

• *Usefulness to You/Your Organization*

☐ Very Useful ☐ Moderately Useful ☐ N/A

• *Did/Will your organization implement the results of this project?* ❑ *Yes* ☐ No

• *How Did You Receive Report?*

☐ Mailed directly to you

☐ Referred to you by someone else

• *Did/Will You Pass Report On To Someone Else*

☐ Yes ☐ No

• *In Your Opinion, Is Anything Missing That Would Make This Report Better?*

☐ Yes _____

_____

. *Genera/Comments*

_____

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 2635 SAN DIEGO CA

POSTAGE WILL BE PAID BY ADDRESSEE

NASSCO/NSRP PROGRAM MANAGER

ATTN: Les Hansen M.S. 20J
National Steel and Shipbuilding Co.
P.O. Box 85278
San Diego, CA 92186-5278

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 2635 SAN DIEGO CA

POSTAGE WILL BE PAID BY ADDRESSEE

NASSCO/NSRP PROGRAM MANAGER

ATTN: Les Hansen M.S. 20J
National Steel and Shipbuilding Co.
P.O. Box 85278
San Diego, CA 92186-5278

Additional copies of this report can be obtained from the National Shipbuilding Research Program Coordinator of the Bibliography of Publications and Microfiche Index. You can call or write to the address or phone number listed below.

NSRP Coordinator
The University of Michigan
Transportation Research Institute
Marine Systems Division
290 Baxter Road
Ann Arbor, Ml 48109-2150

Phone: (313) 763-2465
Fax: (313) 936-1081